Uma Estrutura de Dados Métrica Genérica, Dinâmica, em Memória Secundária

Ângelo Sarmento e Margarida Mamede

CITI, Departamento de Informática
Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa
2829–516 Caparica, Portugal
amlsarmento@gmail.com, mm@di.fct.unl.pt
http://di.fct.unl.pt

Resumo Apresenta-se uma adaptação da estrutura de dados métrica RLC a memória secundária. A RLC é genérica, dinâmica e eficiente quando comparada com outras estruturas de dados métricas implementadas em memória central, onde apenas se contabiliza o número de distâncias calculadas. Neste trabalho, os testes experimentais comparam a RLC com três estruturas de dados implementadas em memória secundária, abrangem dicionários e conjuntos de imagens e medem, quer o número de distâncias calculadas, quer o número de operações de entrada e de saída efectuadas. Os resultados mostram que a RLC é muito eficiente em pesquisas por proximidade e muito competitiva em inserções.

Abstract We introduce an adaptation of the RLC metric data structure to secondary memory. RLC is generic, dynamic, and efficient when compared with other metric data structures implemented in main memory, where performance is analysed only in terms of the number of distance computations. In this work, the experimental study compares RLC with three data structures implemented in secondary memory, comprises two dictionaries and two sets of images, and evaluates both the number of distance computations and the number of I/O operations performed. The results show that RLC is very efficient for range queries and very competitive for insertions.

Keywords: Algorithms, data structures, metric spaces, range search.

1 Introdução

Em muitas áreas (como, por exemplo, biologia computacional, sistemas de informação geográfica ou multimédia), é necessário encontrar os objectos que mais se assemelham a um dado objecto. Essa semelhança é traduzida por uma função que calcula a *distância* entre dois objectos, com base nas suas características. Assume-se que, quanto menor for a distância, mais semelhantes são os objectos.

Devido aos formatos complexos dos dados (e.g. vídeos, imagens, sons, impressões digitais ou sequências de ADN) e à elevada quantidade de informação,

INForum 2010 - II Simpósio de Informática, Luís S. Barbosa, Miguel P. Correia (eds), 9-10 Setembro, 2010, pp. 79–90

é crucial que não se calcule a distância entre cada objecto da base de dados e o objecto fornecido na pesquisa, sempre que uma procura é realizada. O objectivo das *estruturas de dados métricas* é minimizar o número de cálculos de distâncias entre objectos efectuados nas operações sobre a base de dados.

Não obstante nos últimos anos terem sido propostas muitas estruturas de dados métricas [10,17], a maioria é para ser implementada em memória central, sofrendo das limitações inerentes ao espaço disponível. Se nos cingirmos às estruturas de dados métricas implementadas em memória secundária, as alternativas existentes são classificadas como *genéricas*, quando suportam qualquer tipo de objectos e qualquer função de distância, ou como *não genéricas*, no caso contrário. As estruturas OP-tree [13] e SDI-tree [14] são exemplos do segundo grupo, aceitando apenas dados vectoriais. Todas as estruturas de dados métricas genéricas e implementadas em memória secundária são *dinâmicas*, ou seja, permitem realizar actualizações ao seu conteúdo após o carregamento inicial dos dados. No entanto, algumas, como a M-tree [2], a Slim-tree [15] e a DF-tree [16], só suportam a operação de inserção de um novo objecto. É possível efectuar inserções e remoções na SM-tree [12] e na D-Index [4].

Este trabalho aborda o problema da pesquisa por proximidade em estruturas de dados métricas genéricas, dinâmicas e implementadas em memória secundária, estudando uma adaptação da estrutura de dados RLC a memória secundária. A RLC, cujo nome por extenso é *Recursive Lists of Clusters*, é uma estrutura de dados métrica genérica, dinâmica (suportando inserções e remoções) e implementada em memória central. Foi proposta em [6], mas a sua definição inicial foi simplificada (para depender de menos um parâmetro) e os novos algoritmos foram analisados em [7], tendo-se provado que o número médio de distâncias calculadas no carregamento de uma base de dados com n objectos é $O(n \log n)$, numa inserção é $O(\log n)$ e numa remoção é $O(\log^2 n)$. Em todos os estudos comparativos efectuados, quer com dados gerados aleatoriamente [6,7], quer com dados reais (dicionários de línguas naturais [8], imagens de rostos [1] e excertos de música [3]), a RLC mostrou ter um óptimo desempenho.

A adaptação da RLC a memória secundária foi um desafio. Por um lado, as estruturas de dados em memória secundária são muito diferentes das implementadas em memória central. A RLC é uma excepção. Portanto, o bom desempenho comparativo da RLC, obtido em trabalhos anteriores, poderia não se manter. Por outro lado, como a avaliação das estruturas de dados métricas em memória secundária também depende do número de leituras e do número de escritas em ficheiro, para além do número de distâncias calculadas, era necessário alterar o desenho da implementação e os resultados poderiam não ser satisfatórios.

O artigo está organizado da seguinte forma. Na Secção 2, introduzem-se as definições básicas e, na Secção 3, descreve-se brevemente a RLC e a sua adaptação para memória secundária. Depois, na Secção 4, caracterizam-se os espaços métricos usados nos testes experimentais, que são baseados em dicionários e conjuntos de imagens. Na Secção 5, analisam-se os resultados experimentais, que comparam a RLC com três estruturas de dados. Finalmente, a Secção 6 contém alguns comentários ao trabalho desenvolvido e tópicos para trabalho futuro.

2 Definições Básicas

A noção de semelhança ou proximidade entre objectos baseia-se no conceito formal de espaço métrico.

Seja $(\mathcal{U}, d)$ um *espaço métrico*. Ou seja, $\mathcal{U}$ é o *universo* dos *pontos* ou *objectos* e $d : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ é uma função real, chamada *distância* ou *métrica*, que satisfaz as seguintes propriedades, para quaisquer $x, y, z \in \mathcal{U}$:

- (não negatividade) $d(x, y) \geq 0$;
- (identidade) $d(x, y) = 0 \Leftrightarrow x = y$;
- (simetria) $d(x, y) = d(y, x)$; e
- (desigualdade triangular) $d(x, y) \leq d(x, z) + d(z, y)$.

Uma *base de dados sobre* $(\mathcal{U}, d)$ é um conjunto finito $B \subseteq \mathcal{U}$.

Por exemplo, para qualquer $k \geq 1$, os pares $(\mathbb{R}^k, d_1)$ e $(\mathbb{R}^k, d_2)$ são espaços métricos, onde d_1 é a distância de *Manhattan* e d_2 é a distância *euclidiana*:

$$d_1((p_1, p_2, \ldots, p_k), (q_1, q_2, \ldots, q_k)) = \sum_{i=1}^{k} |p_i - q_i| \ , \tag{1}$$

$$d_2((p_1, p_2, \ldots, p_k), (q_1, q_2, \ldots, q_k)) = \sqrt{\sum_{i=1}^{k} (p_i - q_i)^2} \ . \tag{2}$$

Quando o universo é composto por palavras (i.e., sequências de caracteres), é frequente recorrer-se à distância de *Levenshtein*, que indica o número mínimo de operações de edição necessárias para transformar uma palavra na outra. Cada operação de edição pode ser a inserção de um carácter, a remoção de um carácter ou a substituição de um carácter por outro. Formalmente, sejam $X = x_1\, x_2\, \cdots\, x_m$ e $Y = y_1\, y_2\, \cdots\, y_n$ duas palavras (com $m, n \geq 1$). A distância entre X e Y é dada por $d_L(X, Y) = d_{X,Y}(m, n)$, sendo a segunda função definida recursivamente da seguinte forma, onde $\mathrm{dif}(a, b)$ é uma função que vale 0, quando $a = b$, e vale 1, no caso contrário.

$$d_{X,Y}(i, j) = \begin{cases} i, & \text{se } i \geq 0 \text{ e } j = 0; \\ j, & \text{se } i = 0 \text{ e } j > 0; \\ \min(\ d_{X,Y}(i-1, j-1) + \mathrm{dif}(x_i, y_j), & \\ \qquad 1 + d_{X,Y}(i, j-1), & \\ \qquad 1 + d_{X,Y}(i-1, j)\ ), & \text{se } i > 0 \text{ e } j > 0. \end{cases} \tag{3}$$

Dadas uma base de dados B, sobre um espaço métrico $(\mathcal{U}, d)$, e uma *interrogação* (q, r), onde $q \in \mathcal{U}$ e $r \geq 0$, o *problema da pesquisa por proximidade* consiste em encontrar todos os objectos de B cujas distâncias a q não excedem r, i.e., $\{x \in B \mid d(x, q) \leq r\}$. Chama-se a q o *ponto da interrogação* e a r o *raio da interrogação*.

O principal objectivo das estruturas de dados métricas [10,17] é minimizar o número de cálculos de distâncias entre objectos executados durante as pesquisas. Apesar das suas muitas diferenças, todas elas se baseiam na simetria e

na desigualdade triangular para incluir e excluir do conjunto resposta alguns objectos da base de dados, sem calcular as respectivas distâncias ao ponto da interrogação.

3 Recursive Lists of Clusters

Nesta secção define-se a RLC, apresentam-se resumidamente os algoritmos de inserção e de pesquisa (por proximidade) e descrevem-se os aspectos mais importantes da sua implementação em memória secundária.

3.1 Definição da RLC

Basicamente, a RLC (*Recursive Lists of Clusters*) guarda os objectos da base de dados numa lista (ou sequência) de agrupamentos. Cada *agrupamento* (*cluster*) possui:

- um *centro* c, que é um objecto da base de dados;
- um *raio* r, que é um número real não negativo; e
- um *interior* I, que contém um conjunto de objectos da base de dados cujas distâncias a c pertencem ao intervalo $]0, r]$. (Note que a exclusão de zero impede que o centro pertença ao interior do agrupamento.)

A lista de agrupamentos satisfaz uma propriedade fundamental: cada objecto x da base de dados está guardado no primeiro agrupamento que o pode conter, ou seja, se a lista for iterada, x pertence ao primeiro agrupamento (c, r, I) que verificar $d(x, c) \leq r$. Para completar a definição de RLC [7], é necessário especificar dois parâmetros: um real positivo ρ e um inteiro positivo α. O primeiro determina o raio de todos os agrupamentos da lista e o segundo permite definir a implementação dos interiores. Sempre que o número de objectos no interior de um agrupamento não exceder α, o interior é designado por *folha* e está implementado num vector. Nos restantes casos, o interior é uma RLC (com raio ρ e folhas com capacidade α).

Repare que, como o interior de um agrupamento pode ser uma lista de agrupamentos, um objecto pode pertencer a vários agrupamentos, organizados hierarquicamente, aos quais se atribui uma *profundidade*. Considera-se que os agrupamentos da lista principal têm profundidade zero.

Para minimizar o número de distâncias calculadas nas pesquisas (como veremos a seguir), guarda-se, para cada objecto, uma sequência com as distâncias do objecto aos centros dos agrupamentos a que ele pertence, chamada a *sequência de distâncias* do objecto. Portanto, o centro de um agrupamento de profundidade p está associado a p distâncias e cada objecto de uma folha que é o interior de um agrupamento de profundidade p tem uma sequência com $p+1$ distâncias.

A Fig. 1 esquematiza uma RLC com raio 3 e folhas com capacidade 5. Os símbolos c, r, p, $|I|$ e I denotam, respectivamente, o centro, o raio, a profundidade, o número de objectos no interior e o interior do agrupamento. Por exemplo, um objecto w no interior do agrupamento de centro z_1 pertence a três agrupamentos, cujos centros são x_1, y_2 e z_1. A sequência de distâncias de w é constituída por $d(w, x_1)$, $d(w, y_2)$ e $d(w, z_1)$.

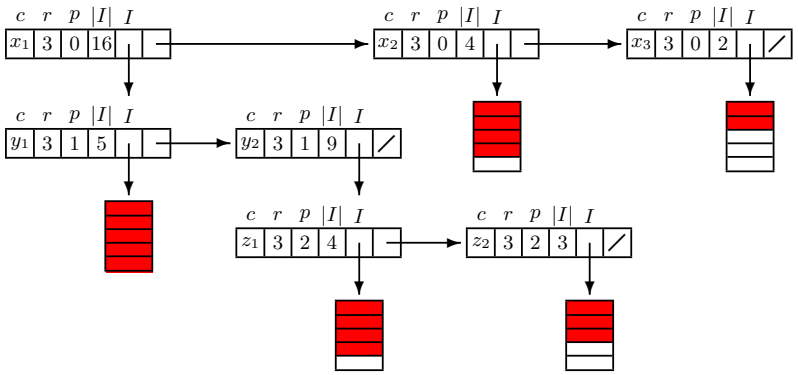

Figura 1. Exemplo de RLC com raio 3 e folhas com capacidade 5. Indica-se o raio e a profundidade de cada agrupamento para clarificar estas noções. Mas, para simplificar a figura, omitem-se todas as sequências de distâncias.

3.2 Descrição Sucinta dos Algoritmos

O próximo objectivo é descrever os grandes passos dos algoritmos sobre a RLC, estando os detalhes especificados em [7].

O carregamento inicial da RLC é feito inserindo sucessivamente cada um dos objectos. Para inserir um objecto x (que, para simplificar, se assume não se encontra na RLC), a lista é iterada até se encontrar um agrupamento (c, ρ, I) que possa conter x (i.e., tal que $d(x, c) \leq \rho$).

- Se nenhum agrupamento for encontrado, cria-se um novo agrupamento (com centro x e interior vazio), que é adicionado à cauda da lista.
- No caso contrário, x é inserido em I.
  - Quando I é uma folha, se há espaço no vector para mais um elemento, x é aí inserido. Se o vector está cheio, transforma-se I numa RLC, criando uma lista de agrupamentos vazia, onde se insere x e cada um dos elementos presentes na folha.
  - Se I é uma lista de agrupamentos, insere-se (recursivamente) x em I.

A remoção é semelhante. Mas não será explicada, por falta de espaço, uma vez que os testes experimentais aqui descritos não envolvem remoções.

Na pesquisas por proximidade, itera-se a lista, identificando-se a relação entre cada agrupamento (c, ρ, I) e a interrogação (q, r), com base na distância entre c e q e no valor dos raios ρ e r. Escusado será dizer que c pertence ao conjunto resposta se $d(c, q) \leq r$. Existem basicamente quatro situações distintas.

- Se a região da interrogação está contida na região do agrupamento, a pesquisa prossegue pelo interior do agrupamento e a iteração pára.
- Se a região da interrogação contém a região do agrupamento, todos os objectos do interior do agrupamento são imediatamente adicionados ao conjunto resposta e a iteração continua.

– Se as regiões da interrogação e do agrupamento se intersectam (mas nenhuma contém a outra), a pesquisa prossegue pelo interior do agrupamento e a iteração continua.
– Se as regiões da interrogação e do agrupamento são disjuntas, o interior do agrupamento é ignorado e a iteração continua.

Repare que se podem incluir ou excluir do conjunto resposta todos os objectos do interior de um agrupamento, sem se ter calculado qualquer distância entre eles e o ponto da interrogação. Quando a pesquisa chega a uma folha, também é possível concluir que um objecto x deve, ou não deve, ser colocado no conjunto resposta, sem se calcular a sua distância a q. Para isso, usam-se as distâncias $d(c, q)$ e $d(x, c)$, onde c é o centro de um agrupamento ao qual x pertence. Fazendo $m = d(c, q) - r$, prova-se que:

– se $d(x, c) < m$, então $d(x, q) > r$ (e x não pertence à resposta);
– se $d(x, c) \leq -m$, então $d(x, q) \leq r$ (e x pertence à resposta).

Note que $d(c, q)$ teve de ser calculado para se entrar no interior do agrupamento e que $d(x, c)$ é um dos elementos da sequência de distâncias de x. A distância entre x e q só é calculada quando, para todo o agrupamento a que x pertence, nenhuma das duas regras pode ser aplicada.

3.3 Implementação em Memória Secundária

A RLC foi implementada em C++.[1] Para reduzir o número de acessos a ficheiro, a informação está guardada em páginas de dimensão fixa (por exemplo, com 4 096 ou 8 192 bytes). Há, basicamente, três tipos de páginas.

– O *cabeçalho* tem informação global da estrutura e do tipo de objectos.
– As *páginas de agrupamentos* permitem implementar as listas de agrupamentos através de listas ligadas de vectores de agrupamentos. Por cada agrupamento, guarda-se o centro, a sequência das distâncias do centro, o número de pontos no interior do agrupamento e o apontador para a (primeira) página onde se encontra o interior do agrupamento.
– De modo semelhante, as *páginas de folhas* permitem implementar cada folha através de uma lista ligada de vectores de pares, cuja primeira coordenada é um objecto e a segunda é a respectiva sequência de distâncias.

Existem três restrições adicionais: não há vectores (de agrupamentos ou de pares) vazios; não existem posições vazias entre os elementos presentes num vector; e nenhum elemento de um vector pode ocupar mais do que uma página.

Esta última restrição impediu que se usassem páginas de dimensão razoável com algumas bases de dados, porque a RLC atingia profundidades tão elevadas que os pares das folhas mais profundas ocupavam demasiada memória. A grande profundidade da RLC deve-se aos raios dos agrupamentos serem todos iguais, como se ilustra na Fig. 2, com pontos no plano e a distância euclidiana.

[1] A primeira implementação da RLC em memória secundária, que só aceitava pontos em $\mathbb{R}^n$ com a distância euclidiana, foi realizada por Carlos Rodrigues [9].

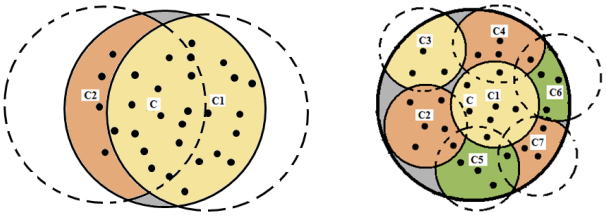

Figura 2. Interior de um agrupamento com centro c. *(Esquerda)* O raio dos agrupamentos é fixo. *(Direita)* O raio dos agrupamentos diminui com a profundidade.

Quando um agrupamento tem muitos pontos, o seu interior I é uma lista de agrupamentos. Ora, quaisquer que sejam os centros dos primeiros agrupamentos dessa lista (c_1 e c_2, do lado esquerdo da Fig. 2), as regiões que eles definem são tão grandes que contêm quase todos os objectos de I. De facto, quando o raio dos agrupamentos é fixo, as listas de agrupamentos (de profundidade positiva) têm um comprimento muito reduzido e a estrutura tem uma profundidade elevada. A ideia da segunda implementação da RLC em memória secundária [11] consiste na redução progressiva do raio dos agrupamentos, à medida que a profundidade aumenta (como exemplificado do lado direito da Fig. 2). Mais precisamente, a RLC continua a ter apenas dois parâmetros, o raio ρ e a capacidade α das folhas, mas o raio de cada agrupamento depende da profundidade p do agrupamento, sendo dado por $\frac{\rho}{p+1}$. Nesta variante, a distribuição dos objectos de um interior pelos agrupamentos do nível seguinte, não só envolve mais agrupamentos, como também é bastante mais equilibrada. Consequentemente, a RLC cresce significativamente em largura, mantendo profundidades muito baixas.

4 Espaços Métricos

O maior desafio das estruturas de dados métricas é terem bons desempenhos com qualquer espaço métrico. O problema é que as distâncias entre os objectos de um universo podem ter distribuições muito diferentes e essa distribuição tem impacto na forma das estruturas de dados e na eficiência dos algoritmos.

Foram seleccionados quatro espaços métricos com dados reais. Dois universos são dicionários[2], tendo sido usada a distância de Levenshtein (3). O dicionário de alemão tem 74 916 palavras, cujos comprimentos variam entre um e trinta e três, e o dicionário de inglês tem 69 069 palavras com comprimentos entre um e vinte e um. O terceiro espaço métrico é composto por histogramas de imagens[3] e pela distância euclidiana (2). Cada um dos 112 543 histogramas é uma sequência de cento e doze números reais. O último universo (cedido por Pedro Chambel [1]) é constituído por 3 040 vectores, cada um com vinte e quatro valores reais que sintetizam características extraídas de imagens de faces humanas. A métrica para as imagens de rostos é a de Manhattan (1).

[2] Os ficheiros com as palavras foram obtidos em `http://www.sisap.org/` .
[3] O ficheiro foi retirado de `http://www.dbs.informatik.uni-muenchen.de/` .

Para conhecer algumas propriedades dos espaços métricos, calcularam-se, para cada universo, todas as distâncias entre dois objectos distintos. A Fig. 3 apresenta os histogramas com as distribuições dessas distâncias e mais algumas estatísticas. No caso dos dois espaços métricos de imagens, como as distâncias são reais, foram agrupadas em vinte e cinco intervalos de igual dimensão.

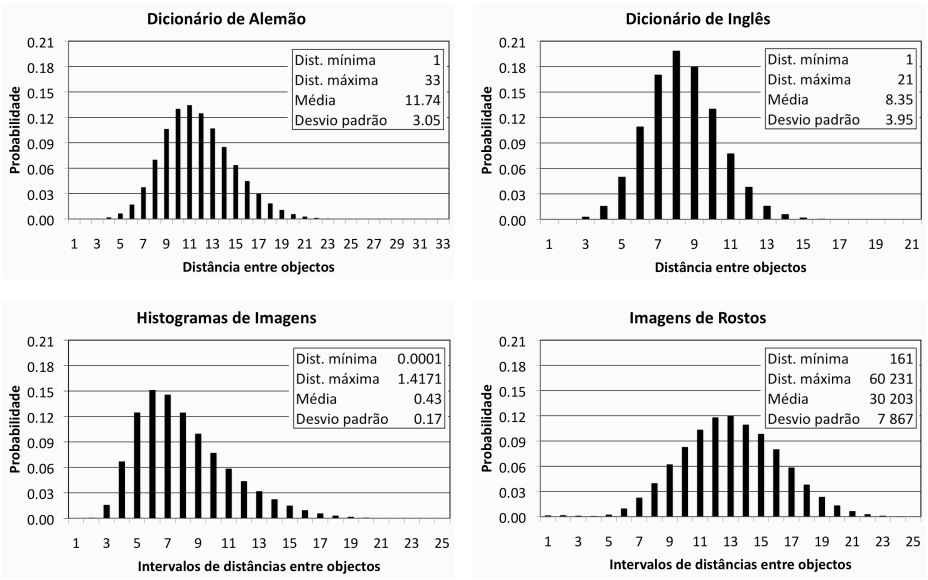

Figura 3. Histogramas das distâncias para os quatro espaços métricos.

5 Resultados Experimentais

A RLC foi comparada com três estruturas de dados métricas genéricas, implementadas em memória secundária: M-tree [2], Slim-tree [15] e DF-tree [16]. Usaram-se as implementações da biblioteca GBDI Arboretum [5], escrita em C++. Não se testaram mais estruturas de dados devido à inexistência de implementações de domínio público.

Todas as estruturas de dados são parametrizadas. Para o parâmetro comum, que é a dimensão das páginas do ficheiro em bytes, foi escolhido o valor 4 096. Em relação à M-tree, usaram-se os métodos mais eficazes: o do hiperplano generalizado para particionar os nós e o que minimiza a soma dos raios para efectuar as promoções. A implementação da Slim-tree permite definir a sub-árvore onde se insere um objecto, quando mais do que uma o pode conter, e o método de particionamento dos nós. As escolhas, aconselhadas em [15], recaíram, respectivamente, sobre o método que selecciona a sub-árvore cuja raiz tem menor ocupação e sobre o particionamento induzido pela árvore mínima de cobertura. A DF-tree

tem quatro parâmetros, dos quais dois são os da Slim-tree, que seleccionaram os mesmos algoritmos. Para o número de representantes globais, recorreu-se à dimensão máxima dos pontos do universo: 33 para o dicionário de alemão, 21 para o de inglês, 112 para os histogramas de imagens e 24 para as imagens de rostos. Por último, o valor a partir do qual o conjunto de representantes globais é actualizado foi o utilizado pelos autores nos seus testes experimentais [16]: 2 000. Os parâmetros da RLC, determinados por observação de resultados experimentais [11], variam com o espaço métrico. O raio usado foi 10, 6, 0.71 e 23 160, e a capacidade das folhas foi 100, 100, 50 e 25, respectivamente, para os dicionários de alemão e de inglês, os histogramas de imagens e as imagens de rostos.

Das quatro estruturas de dados, a RLC é a única que suporta a operação de remoção. Por esse motivo, os testes experimentais relatados neste artigo só avaliam inserções e pesquisas (por proximidade).

Foram gerados quatro ficheiros por cada espaço métrico. Três são permutações aleatórias do universo e foram usados para carregar a base de dados por ordens diferentes. O outro, com cerca de 10% dos objectos do domínio (também seleccionados aleatoriamente), contém as interrogações. Os raios das interrogações foram gerados uniformemente dentro do intervalo $[1, 3]$ para os dicionários, $[0.00001, 0.1]$ para os histogramas de imagens e $[5\,000, 15\,000]$ para as imagens de rostos. Para cada espaço métrico, cada teste consistiu exactamente nas mesmas inserções e nas mesmas pesquisas, variando apenas a ordem pela qual os objectos foram inseridos, que tem influência na forma final das estruturas de dados. Todos os resultados apresentados são a média dos valores obtidos nos três testes do mesmo universo.

A Fig. 4 apresenta o número médio de distâncias calculadas por operação de inserção e de pesquisa, o número médio de leituras de ficheiro efectuadas por operação de inserção e de pesquisa, e o número médio de operações de escrita em ficheiro por operação de inserção. Para os valores das distâncias, dividiram-se os respectivos números médios pelo número de objectos do universo, para se compreender melhor o desempenho das estruturas de dados. Repare que, se essa percentagem for 70% (que é o valor para a pesquisa na Slim-tree com o dicionário de inglês), o ganho face ao algoritmo que percorre um vector com os objectos da base de dados e calcula as distâncias entre todos os elementos do vector e o ponto da interrogação é de apenas 30%.

Numa primeira análise, pode-se concluir que os valores mínimos estão sempre associados à M-tree, à Slim-tree ou à RLC. De facto, a RLC não tem o melhor desempenho em apenas cinco dos vinte casos: no número de distâncias por inserção com os dicionários de alemão (0.12% na Slim-tree e 0.50% na RLC) e de inglês (0.14% na Slim-tree e 0.36% na RLC) e no número de leituras por inserção com o dicionário de alemão (3 na M-tree e na Slim-tree; 6 na RLC) com o dicionário de inglês (2.9 na M-tree e na Slim-tree; 3.4 na RLC) e com os histogramas de imagens (11 na Slim-tree e 15 na RLC). No entanto, sempre que os valores da RLC não são os mais baixos, as diferenças são pouco significativas. Em todos os outros casos, e, em particular, nos dois parâmetros da avaliação das pesquisas e no número de escritas em ficheiro, a RLC é imbatível.

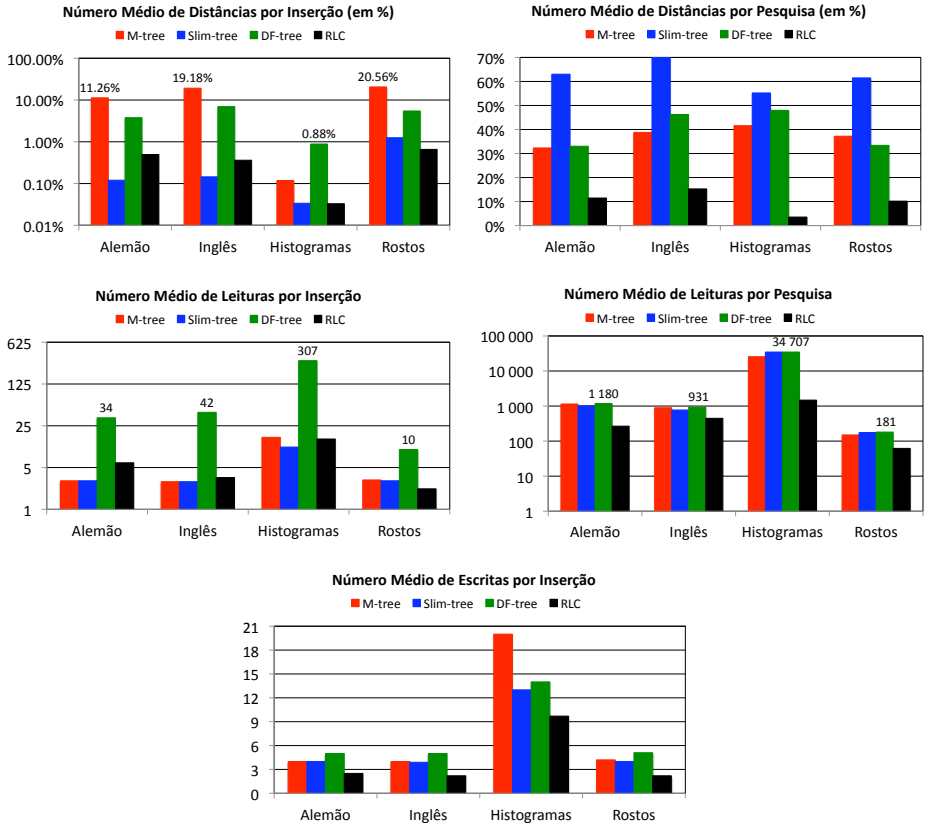

Figura 4. Números médios de distâncias calculadas entre objectos, de leituras e de escritas em ficheiro, por inserção e por pesquisa. Nos dois gráficos de cima, os valores correspondem à percentagem do número médio de distâncias calculadas em relação ao número de objectos da base de dados. Note que a escala de três gráficos é exponencial, para que todas as barras sejam visíveis. Nesses casos, apresenta-se o valor da barra mais alta para cada espaço métrico.

Estes resultados ainda são mais favoráveis para a RLC do que aqueles que se obtiveram ao comparar estruturas de dados métricas em memória central [6,7,8,1,3]. Note que as diferenças nos números de distâncias calculadas nas pesquisas são impressionantes. Acresce que a RLC é verdadeiramente dinâmica. Portanto, ao contrário do que é usual quando se comparam estruturas de dados, neste caso não é necessário optar entre um bom desempenho nas pesquisas e nas inserções e a possibilidade de se efectuar uma operação tão importante como a remoção.

Na Tabela 1, apresentam-se alguns dados sobre a forma da RLC, obtidos com um dos ficheiros de teste. Verifica-se que há interiores implementados em listas com muitos agrupamentos e que a profundidade da estrutura é pequena.

Tabela 1. Dados sobre a forma da RLC com todos os objectos do espaço métrico.

Espaço métrico	Número de agrupamentos	Comprimento da lista principal	Comprimento máximo de outra lista	Profundidade máxima
Alemão	22 976	2 360	2 494	4
Inglês	18 559	2 281	1 033	4
Histogramas	25 747	91	422	9
Rostos	326	42	67	2

6 Conclusões e Trabalho Futuro

Descreveu-se e avaliou-se uma adaptação da estrutura de dados RLC a memória secundária. A RLC revelou-se equivalente à Slim-tree nas inserções, com a qual partilhou os melhores desempenhos, e consideravelmente mais eficiente que todas as outras estruturas de dados analisadas nas pesquisas por proximidade.

Em relação ao trabalho futuro, pretende-se continuar a avaliar a RLC, estendendo os testes experimentais a dados de outras áreas (como biologia computacional ou sistemas de informação geográfica) e a mais implementações de estruturas de dados métricas. É importante obter versões da Slim-tree e da DF-tree que incorporem o algoritmo *slim-down* [15] (que não está implementado na biblioteca Arboretum) e implementações de estruturas de dados que suportem a operação de remoção.

Para minorar o problema da parametrização da RLC, pretende-se descobrir fórmulas para o raio e para a capacidade das folhas que dependam de características do espaço métrico. Como o impacto do raio no desempenho da estrutura é enorme, e o impacto da capacidade das folhas é bastante reduzido, começou-se pelo primeiro, tendo-se chegado à fórmula $\rho = \mu - \frac{\sigma}{2}$, onde μ é a média e σ é o desvio padrão da distribuição das distâncias. Parece que os valores obtidos (c.f. Tabela 2) são uma boa aproximação para distribuições normais, não se aplicando à distribuição dos histogramas de imagens.

Tabela 2. Valores da fórmula para o raio, para distribuições normais.

	Alemão	Inglês	Histogramas	Rostos
Média (μ)	11.74	8.35	0.43	30 203
Desvio padrão (σ)	3.05	3.95	0.17	7 867
Raio usado	10	6	0.71	23 160
Valor de $\mu - \frac{\sigma}{2}$	10.22	6.38	0.35	26 270

Agradecimentos Ângelo Sarmento teve uma bolsa semestral concedida pelo Centro de Informática e Tecnologias da Informação (CITI), hospedado no Departamento de Informática da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa. Os autores agradecem a Luís Caires, por ter disponibilizado

uma máquina eficiente para a realização dos testes experimentais, a Carlos Rodrigues, pela ajuda na utilização da biblioteca Arboretum, e a Pedro Chambel, pela cedência dos dados das imagens de rostos.

Referências

1. Chambel, P.: Pesquisa de Imagens de Rosto. Master's thesis, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa (2009), http://citi.di.fct.unl.pt/
2. Ciaccia, P., Patella, M., Zezula, P.: M-tree: An efficient access method for similarity search in metric spaces. In: 23rd Int. Conf. on Very Large Data Bases (VLDB'97). pp. 426–435. Morgan Kaufmann, San Francisco (1997)
3. Costa, F.: Geração automática de playlists de músicas semelhantes. Master's thesis, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa (2009), http://citi.di.fct.unl.pt/
4. Dohnal, V., Gennaro, C., Savino, P., Zezula, P.: D-Index: Distance searching index for metric data sets. Multimedia Tools and Applications 21(1), 9–33 (2003)
5. GBDI Arboretum, http://www.gbdi.icmc.usp.br/arboretum/
6. Mamede, M.: Recursive lists of clusters: A dynamic data structure for range queries in metric spaces. In: 20th Int. Symp. on Computer and Information Sciences (ISCIS 2005). LNCS, vol. 3733, pp. 843–853. Springer, Heidelberg (2005)
7. Mamede, M.: A dynamic data structure for range queries in high dimensional metric spaces. Tech. rep., Departamento de Informática, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa (2007), http://ctp.di.fct.unl.pt/~mm/dynamic-07.pdf
8. Mamede, M., Barbosa, F.: Range queries in natural language dictionaries with recursive lists of clusters. In: 22nd Int. Symp. on Computer and Information Sciences (ISCIS 2007). pp. 1–6. IEEE CS Press, Los Alamitos, CA (2007)
9. Rodrigues, C.: Implementação de sistemas de indexação para espaços métricos. Relatório do projecto final de curso, Departamento de Informática, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa (2006)
10. Samet, H.: Foundations of Multidimensional and Metric Data Structures. Computer Graphics and Geometric Modeling, Morgan Kaufmann, San Francisco (2006)
11. Sarmento, A.: Estruturas de Dados Métricas Genéricas em Memória Secundária. Master's thesis, Departamento de Informática, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa (2010), http://citi.di.fct.unl.pt/
12. Sexton, A.P., Swinbank, R.: Symmetric M-tree. Tech. Rep. CSR-04-02, School of Computer Science, University of Birmingham (2004), http://www.cs.bham.ac.uk/~rjs/research/
13. Thomasian, A., Zhang, L.: Persistent semi-dynamic ordered partition index. The Computer Journal 49(6), 670–684 (2006)
14. Thomasian, A., Zhang, L.: The stepwise dimensionality increasing (SDI) index for high-dimensional data. The Computer Journal 49(5), 609–618 (2006)
15. Traina, Jr., C., Traina, A., Faloutsos, C., Seeger, B.: Fast indexing and visualization of metric data sets using Slim-trees. IEEE Transactions on Knowledge and Data Engineering 14(2), 244–260 (2002)
16. Traina, Jr., C., Traina, A., Filho, R.S., Faloutsos, C.: How to improve the pruning ability of dynamic metric access methods. In: 11th Int. Conf. on Information and Knowledge Management (CIKM'02). pp. 219–226. ACM Press, New York (2002)
17. Zezula, P., Amato, G., Dohnal, V., Batko, M.: Similarity Search: The Metric Space Approach. Advances in Database Systems, Springer (2006)