Self-organizing Data Structures with Dependent Accesses

Frank Schulz and Elmar Schömer

Universität des Saarlandes, FB 14, Informatik, Lehrstuhl Prof. Dr. G. Hotz,
Postfach 151150, 66041 Saarbrücken, Germany.
Email: {schulz,schoemer}@cs.uni-sb.de

Abstract. We consider self-organizing data structures in the case where the sequence of accesses can be modeled by a first order Markov chain. For the simple-k- and batched-k–move-to-front schemes, explicit formulae for the expected search costs are derived and compared. We use a new approach that employs the technique of expanding a Markov chain. This approach generalizes the results of Gonnet/Munro/Suwanda.
In order to analyze arbitrary memory-free move-forward heuristics for linear lists, we restrict our attention to a special access sequence, thereby reducing the state space of the chain governing the behaviour of the data structure.
In the case of accesses with locality (inert transition behaviour), we find that the hierarchies of self-organizing data structures with respect to the expected search time are reversed, compared with independent accesses.
Finally we look at self-organizing binary trees with the move-to-root rule and compare the expected search cost with the entropy of the Markov chain of accesses.

1 Introduction and summary

We consider the dictionary problem when the query source can be modelled by an ergodic Markov chain. In (Hotz [7]) it has been shown that a search graph of size $\Theta(n^2)$ can be constructed such that the expected search time is asymptotically equal to the entropy of the Markov chain. When the transition behaviour of the chain exhibits the phenomenon of locality, that expected search time can also be achieved with $O(n)$ memory.

Here we want to continue that line of research. We examine approximation schemes that are restricted to use only linear amounts of memory. In literature, these techniques are known as self-organizing data structures.

Consider a linear list containing the elements $1, 2, \ldots, n$. At each access the list has to be searched sequentially, and the search cost is assumed to be proportional to the number of comparisons needed to find the requested element.

Heuristics for self-organization perform a reordering of the list and thereby hope to reduce the search cost for the next accesses. An example is the move-to-front rule (MTF) which places the requested element at the head of the list and slides the other elements back one position. Another memory-free heuristic is the transposition rule (TR) which moves the accessed element one place towards the head of the list.

1.1 Known results for independent accesses

In the case of independent accesses, the requests are assumed to follow a fixed but unknown probability distribution $(p_1, \ldots, p_n)$ such that element i is accessed with probability p_i. Since no ordering is used for sequential search, we may assume $p_1 \geq \cdots \geq p_n$. When the list is ordered by decreasing access probabilities, the expected search cost is minimized, yielding $M = \sum_{i=1}^{n} i p_i$. Let $b(j,i)$ denote the asymptotic probability that element j stands in front of element i in the list when the move-to-front heuristic is used. It is easy to see that for $i \neq j$, we have $b(j,i) = p_j/(p_i + p_j)$. It has been shown that the expected search cost under the move-to-front rule is

$$\mu \;=\; \sum_i p_i \cdot \big(1 + \sum_{j \neq i} b(j,i)\big) \;=\; 1 + 2 \sum_{i<j} \frac{p_i p_j}{p_i + p_j} \;\leq\; \frac{\pi}{2} \cdot M \;\;,$$

(see Burville/Kingman [2], Chung/Hajela/Seymour [4]).

A class of heuristics that are allowed to use some memory consists of the k-in-a-row strategies, in combination with a memory-free rule R (Gonnet/Munro/Suwanda [5]). With the simple-k-heuristic, the action of R is performed only if the same element has been requested k times in a row. The batched-k-heuristic groups the accesses into blocks of length k and calls R only if all requests within the block are the same. The k-in-a-row strategies use $O(\log k + \log n)$ memory in order to store the counter value between 1 and k and the element requested last. Let $b_k(j,i)$ and $b'_k(j,i)$ denote the asymptotic probability that j stands in front of i in the list when the simple-k-move-to-front rule or the batched-k-move-to-front rule respectively are applied. It has been shown that

$$b_k(j,i) = \frac{p_j^k \sum_{l=0}^{k-1} p_i^l}{p_j^k \sum_{l=0}^{k-1} p_i^l + p_i^k \sum_{l=0}^{k-1} p_j^l} \tag{1}$$

$$b'_k(j,i) = \frac{p_j^k}{p_i^k + p_j^k} \;\;, \tag{2}$$

which allows for the calculation of the expected search time μ. It has been demonstrated that considerable improvements of the search time are achieved even for small values of k, and that $\mu \to M$ as $k \to \infty$. Furthermore, for fixed k batched-k-MTF performs better than simple-k-MTF (see Gonnet/Munro/Suwanda [5]).

The move-to-front heuristic is an example from the class of memory-free move-forward rules. Such a rule is specified by a sequence $1 = m_1 \leq m_2 \leq \cdots \leq m_n$ with $m_i < i$ for all $i = 2, \ldots, n$. The interpretation is that an element at position i of the list is moved to position m_i upon request, leaving the relative ordering of the other elements unchanged. Examples of these rules are:

MOVE-UP(k) the requested element is placed at the head of the list if its position is $i \leq k$, and moved k places up if its position is $i > k$

POS(k) the requested element is moved one place up if its position is $i \leq k$, and moved to position k if its position is $i > k$

SWITCH(k) the accessed element is moved to the head of the list if its position is $i \leq k$, and moved one place up if its position is $i > k$.

A partial ordering of the move-forward rules can be defined as follows: given two rules R and R' specified by $m_1, \ldots, m_n$ and $m'_1, \ldots, m'_n$ respectively, $R \leq R'$ holds if $m'_i \leq m_i$ for all $i = 1, \ldots, n$. The following spectra of rules can be compared:

$$\begin{aligned}
&\text{TR} \leq R \leq \text{MTF} \quad \text{for all move-forward rules } R \\
&\text{MOVE-UP}(k) \leq \text{MOVE-UP}(k+1) \\
&\text{POS}(k+1) \leq \text{POS}(k) \\
&\text{SWITCH}(k) \leq \text{SWITCH}(k+1)
\end{aligned}$$

For the special access distribution $p_1 = \alpha$, $p_2 = \cdots = p_n = \beta$, it has been shown by Lam [12] that the expected search times $\mu(R)$ and $\mu(R')$ of two move-forward rules R and R' obey

$$R \leq R' \implies \mu(R) \leq \mu(R') \ , \tag{3}$$

which entails the corresponding relations of expected search time for the spectra described above (see Kan/Ross [8], Lam [12], Phelps/Thomas [14], Tenenbaum/Nemes [16]).

1.2 Known results for dependent accesses

In the case of dependent accesses we assume that the sequence of requests can be modeled by an ergodic first-order Markov chain with transition matrix $P = (p_{ij})$ and stationary distribution $(q_1, \ldots, q_n)$. Let $M = (m_{ij})$ denote the matrix of mean first-passage times. It has been shown that the asymptotic probability $b(j, i|i)$ that j stands in front of i in the list when i is requested and the move-to-front rule is given by $b(j, i|i) = 1/(q_i(m_{ij} + m_{ji}))$, and hence the expected search time is

$$\mu \ = \ \sum_{i=1}^{n} q_i \cdot \big(1 + \sum_{j \neq i} b(j, i|i)\big) \ = \ 1 + 2 \sum_{i<j} \frac{1}{m_{ij} + m_{ji}}$$

(see Lam/Leung/Siu [13]). Since the heuristic does not have a memory, a query source with memory can always outperform it. There are situations, however, in which the heuristic performs very well. These situations can be described by the phenomenon of *locality*: a small subset of elements is requested for a long time before the access sequence switches to another small subset.

Recently it has been demonstrated that in some of these cases, the move-to-front heuristic is optimal in that its expected search time is not greater than the expected search time of any other sequential search strategy, even if that search strategy may use arbitrary amounts of memory (Chassaing [3]).

1.3 New results

We consider the simple-k- and batched-k-move-to-front schemes and derive formulae for the asymptotic probabilities $b_k(j,i|i)$ and $b'_k(j,i|i)$ that j stands in front of i when i is requested. This is achieved by expanding the chain of accesses to k-tuples and looking for the occurrence of tuples which consist solely of i's or j's. In the special case of independent accesses the results cited above are obtained.
As an example, we calculate the expected search cost for the access sequence defined by

$$p_{ij} = \left\{ \begin{array}{lcl} \alpha & : & i = j \\ \beta & : & i \neq j \end{array} \right\}$$

with $0 \leq \alpha < 1$ and $\beta = (1-\alpha)/(n-1)$. The phenomenon of locality appears for $\alpha > \beta$. In this case, we observe that the expected search time gets worse for increasing k and approaches $(n+1)/2$ as $k \to \infty$. Furthermore, simple-k-MTF performs better than batched-k-MTF. Hence the ordering of strategies with respect to the expected search time is reversed, compared with independent accesses.

As for the analysis of the class of memory-free move-forward rules, we restrict our attention to the same access sequence. This enables us to reduce the state space of the Markov chain governing the behaviour of the data structure. Given two move-forward rules R, R' and $\alpha \geq \beta$, the expected search times satisfy

$$R \leq R' \implies \mu(R') \leq \mu(R) \ .$$

As a consequence, we find again that the hierarchy of heuristics is exactly reversed, compared with independent accesses (see (3)). For the first time, memory-free heuristics other than move-to-front are analyzed in the case of Markovian access sequences.

Finally we look at self-organizing binary search trees, and compare the expected search time with the entropy of the chain of accesses.

2 *k*-in-a-row heuristics

In order to analyze the simple-k-MTF heuristic, we observe

Lemma 1. *Using simple-k-move-to-front, the element j stands in front of i if the last sequence of k subsequent accesses to j has occurred after the last sequence of k subsequent accesses to i.*

At any instant in time, the current form of the data structure depends on its history. Therefore, we want to employ backward analysis and look at the past. Hence we reformulate the lemma:

When l is requested, j stands in front of i if we trace back the chain of accesses starting from l and encounter k subsequent accesses to j before we encounter k subsequent accesses to i.

Two techniques from Markov chain theory are needed: reversal of time and expansion of the chain (see for example Kemeny/Snell [10]).
When P is an ergodic Markov chain with state space $S = \{1, \ldots, n\}$ and stationary distribution $(q_1, \ldots, q_n)$, the time-reversed chain $\hat{P}$ is defined by its transition probabilities

$$\hat{p}_{ij} = \frac{q_j p_{ji}}{q_i} \quad .$$

For $k > 1$, the expanded chain $\tilde{P}$ has the state space

$$\tilde{S} = \{[i_1 \ldots i_k] \mid i_j \in S, p_{i_1 i_2} \cdot \ldots \cdot p_{i_{k-1} i_k} > 0\} \quad .$$

A transition can be viewed as sliding a window of size k over the original sequence:

$$\tilde{p}_{[i_1 \ldots i_k][j_1 \ldots j_k]} = p_{i_k j_k} \cdot \delta_{[i_2 \ldots i_k][j_1 \ldots j_{k-1}]} \quad ,$$

where

$$\delta_{[a_1 \ldots a_j][b_1 \ldots b_j]} = \left\{ \begin{array}{lcl} 1 & : & a_i = b_i \text{ for } i = 1, \ldots, j \\ 0 & : & \text{otherwise} \end{array} \right\}$$

denotes the Kronecker symbol. The fundamental matrix of an ergodic chain is defined as $Z = (z_{ij}) = (I - P + Q)^{-1}$, where Q is the limiting transition matrix of the chain [10]. From this, the first passage times can be calculated as $m_{ij} = (\delta_{ij} - z_{ij} + z_{jj})/q_j$. We want to express the first passage times of the time-reversed expanded chain with quantities of the original chain. This gives:

$$\begin{aligned} \hat{m}_{[i_1 \ldots i_k][j_1 \ldots j_k]} &= z_{j_k j_1}/q_{j_1} - z_{j_k i_1}/q_{i_1} \\ &+ \frac{1 - \sum_{l=1}^{k-2} q_{j_1} p_{j_1 j_2} \cdots p_{j_l j_{l+1}} (\delta_{j_{l+1} i_1} \cdots \delta_{j_k i_{k-l}}/q_{i_1} - \delta_{j_{l+1} j_1} \cdots \delta_{j_k j_{k-l}}/q_{j_1})}{q_{j_1} p_{j_1 j_2} \cdots p_{j_{k-1} j_k}} \quad . \end{aligned}$$

For an ergodic chain, the probability of starting in state l and reaching state j without passing through the taboo state i is given by

$${}_i f^*_{lj} = \frac{m_{li} + m_{ij} - m_{lj}}{m_{ij} + m_{ji}}$$

(see Kemperman [11]). Let $b([j \ldots j], [i \ldots i] \mid [l_1 \ldots l_{k-1} i])$ be the asymptotic probability that $[j \ldots j]$ appears before $[i \ldots i]$ when we start the time-reversed expanded chain in state $[l_1 \ldots l_{k-1} i]$. With the above formula, this can be calculated from

$$b([j \ldots j], [i \ldots i] | [l_1 \ldots l_{k-1} i]) = \frac{\hat{m}_{[l_1 \ldots l_{k-1} i][i \ldots i]} + \hat{m}_{[i \ldots i][j \ldots j]} - \hat{m}_{[l_1 \ldots l_{k-1} i][j \ldots j]}}{\hat{m}_{[i \ldots i][j \ldots j]} + \hat{m}_{[j \ldots j][i \ldots i]}} \quad .$$

Given these probabilities, and taking the mean over all $l_1, \ldots, l_{k-1}$, we get the asymptotic probability $b_k(j, i|i)$ that j stands in front of i when i is accessed and the simple-k-move-to-front rule is used. The probability of occurence of $[l_1 \ldots l_{k-1} i]$ is the probability to trace back the sequence from i and encountering $l_{k-1}, \ldots, l_1$ subsequently. Hence

$$b_k(j, i|i) = \sum_{l_1 \ldots l_{k-1}} \hat{p}_{i l_{k-1}} \cdot \hat{p}_{l_{k-1} l_{k-2}} \cdots \hat{p}_{l_2 l_1} \cdot b([j \ldots j], [i \ldots i] | [l_1 \ldots l_{k-1} i]) \quad .$$

Let $p_{ij}^{(t)}$ denote the probability of going from i to j in exactly t steps. Combining these considerations, we get the following result.

Theorem 2.

$$b_k(j,i|i) = \frac{q_j p_{jj}^{k-1}\left[\sum_{t=0}^{k-2} p_{ii}^t + p_{ii}^{k-1}\sum_{t=1}^{k-1}(p_{ji}^{(t)} - p_{ii}^{(t)}) + p_{ii}^{k-1}\Big(q_i m_{ji} + \sum_l q_l p_{li}^{(k-1)}(m_{il} - m_{jl})\Big)\right]}{q_i p_{ii}^{k-1}\sum_{t=0}^{k-2} p_{jj}^t \;+\; q_j p_{jj}^{k-1}\sum_{t=0}^{k-2} p_{ii}^t \;+\; q_i q_j p_{ii}^{k-1} p_{jj}^{k-1}(m_{ij} + m_{ji})}$$

See [15] for the details of the derivation. The expected search time under the simple-k-move-to-front rule is $\mu = 1 + \sum_{j\neq i} q_i \cdot b_k(j,i|i)$.

For independent accesses, $p_{ij} = q_j$, hence $p_{ij}^{(t)} = q_j$ for $t \geq 1$ and $m_{ij} = 1/q_j$, thus formula (1) for $b_k(j,i)$ is obtained.

For the batched-k-move-to-front, we observe

Lemma 3. *Employing batched-k-move-to-front, the element j stands in front of i if the last block of accesses to j has occured after the last block of accesses to i.*

In order to apply backward analysis, we use the reformulation:

When l is requested, j stands in front of i if we trace back the chain of accesses starting from the current block (the block containing l) and encounter a block of j's before a block of i's has appeared.

Simplifying the calculation, we use the transition matrix that combines k steps into one: $\bar{P} = P^k$. Let $\bar{Z} = (\bar{z}_{ij}) = (I - \bar{P} + Q)^{-1}$ be the corresponding fundamental matrix, where Q is the limiting transition matrix of the ergodic chain (Kemeny/Snell [10]) .
Using a similar technique as above, we define an expanded chain with transition probabilities

$$\tilde{p}_{[i_1\ldots i_k][j_1\ldots j_k]} = p_{i_k j_1} \cdot p_{j_1 j_2} \cdots p_{j_{k-1} j_k} \quad .$$

Then we calculate the asymptotic probability $b([j\ldots j],[i\ldots i]|l)$ that $[j\ldots j]$ appears before $[i\ldots i]$ appears, when the time-reversed expanded chain is started from $[l\ldots]$. As no tuples overlap, it is sufficient to know the first element l of the current block.

$$b([j\ldots j],[i\ldots i]|l) = \frac{\hat{m}_{[l\ldots][i\ldots i]} + \hat{m}_{[i\ldots i][j\ldots j]} - \hat{m}_{[l\ldots][j\ldots j]}}{\hat{m}_{[i\ldots i][j\ldots j]} + \hat{m}_{[j\ldots j][i\ldots i]}}$$

In the current block, the current access to i can appear at position $1, 2, \ldots, k$, each with probability $1/k$. When it occurs at position m within the current block, we go $m-1$ steps back to reach the first element of the block. This element is l with probability $\hat{p}_{il}^{(m-1)}$. Hence

$$b'_k(j,i|i) = \frac{1}{k}\sum_{m=1}^{k}\sum_l \hat{p}_{il}^{(m-1)} \cdot b([j\ldots j],[i\ldots i]|l)$$

Theorem 4.

$$b'_k(j,i|i) = \frac{q_j p_{jj}^{k-1} - q_j p_{ii}^{k-1} p_{jj}^{k-1} \sum_x \Big(\bar{z}_{jx} - \bar{z}_{ix}\Big)\Big(p_{xi} - \sum_{m=1}^{k} p_{xi}^{(m)}/k\Big)}{q_i p_{ii}^{k-1} + q_j p_{jj}^{k-1} - p_{ii}^{k-1} p_{jj}^{k-1} \sum_x \Big(\bar{z}_{jx} - \bar{z}_{ix}\Big)\Big(q_j p_{xi} - q_i p_{xj}\Big)}$$

For independent accesses, formula (2) for $b'_k(j,i)$ follows as a corollary.
We want to illustrate the results with the access sequence

$$p_{ij} = \left\{ \begin{array}{lcl} \alpha & : & i = j \\ \beta & : & i \neq j \end{array} \right\} \tag{4}$$

with $0 \leq \alpha < 1$ and $\beta = (1-\alpha)/(n-1)$. For $\alpha > 1/n$, this sequence exhibits the phenomenon of elementwise locality, the degree of locality can be adjusted via the parameter α.

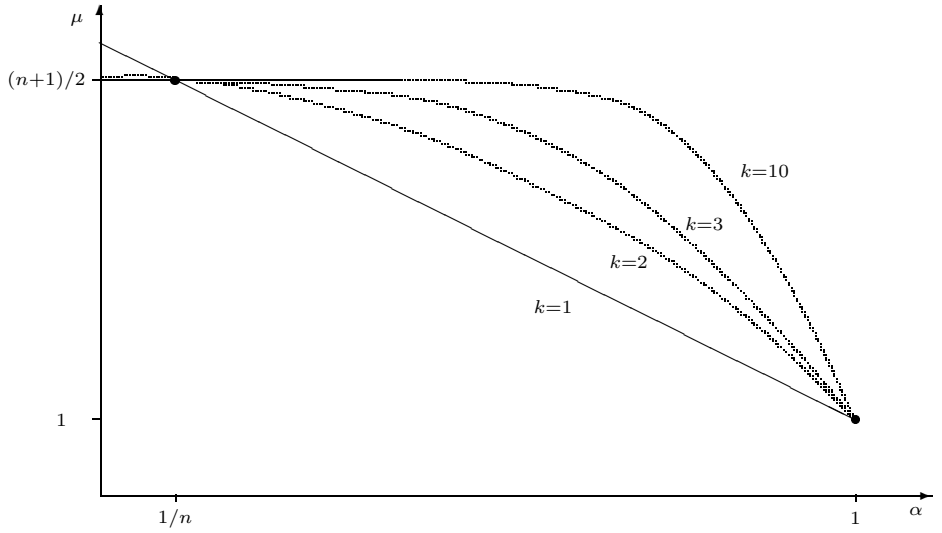
Theorem 5. *For the access sequence (4), the simple-k-move-to-front rule produces the expected search cost*

$$\mu = 1 + \frac{n(n-1)(1-\alpha^k)}{2(n-\alpha^{k-1})} \ ,$$

and the batched-k-move-to-front rule

$$\mu' = 1 + \frac{n-1}{2} \cdot \left(1 - \frac{\alpha^{k-1}(n\alpha-1)\Big((n-1)^k - (n\alpha-1)^k\Big)}{k(n-n\alpha)\Big((n-1)^k - (n\alpha-1)^k + \alpha^{k-1}(n\alpha-1)(n-1)^{k-1}\Big)} \right) \ .$$

One easily sees that for $\alpha > 1/n$ (locality of references), the expected search time is monotone increasing in k. For all $\alpha < 1$, it approaches $(n+1)/2$ as $k \to \infty$. The expected search time in dependence of α looks as follows:



Comparing μ and μ', and using Bernoulli's inequality, we find

Theorem 6. *For the access sequence (4) and all $n \geq 2$, $k \geq 2$, the simple-k rule performs better than the batched-k rule in the situation of locality:*

$$\alpha > \beta \implies \mu < \mu' \ .$$

Again, the order is reversed, compared with independent accesses.

3 Move-forward rules

In order to analyze arbitrary move-forward rules, one has to look at the Markov chain that describes the states of the data structure and the transitions between them. Each permutation of the list is a state, and a transition from state π to state σ is possible if there is an element i such that σ is obtained when the action of the heuristic is applied to π upon request of i.

In the case of independent accesses that are identically distributed with the distribution $(p_1, \ldots, p_n)$, this transition has the probability $p_{\pi\sigma} = p_i$.
When accesses are dependent and governed by the transition matrix $P = (p_{ij})$, we have to consider the extended Markov chain with states $(\pi|i)$, where π is the current permutation of the list and i is the element that will be requested next. Then the transitions have the probability

$$p_{(\pi|i)(\sigma|j)} = \left\{ \begin{array}{ccl} p_{ij} & : & \pi \text{ is permuted to } \sigma \text{ after the request of } i \\ 0 & : & \text{otherwise} \end{array} \right\} \ .$$

In the situation of independent accesses, several authors have restricted the accesses to the distribution $p_1 = \alpha, p_2 = \cdots = p_n = \beta$ (see [8, 12, 14, 16]). The benefit is that it now suffices to know the position of element 1. Thus, the size of the state space can be reduced from $n!$ to n (by lumping the chain).

For dependent accesses, we take a similar approach. By restricting our attention to the access sequence defined by the Markov chain with transition probabilities

$$p_{ij} = \left\{ \begin{array}{ccl} \alpha & : & i = j \\ \beta & : & i \neq j \end{array} \right\} \ , \tag{5}$$

we claim that the size of the state space can be reduced from $nn!$ to n. It is sufficient to know the position of the element that is requested next. Let $q(i_1, \ldots, i_n|i_j)$ be the asymptotic probability that the list is in configuration $\pi = (i_1, \ldots, i_n)$ and i_j is requested next. Then the definition

$$q(j) = n! \cdot q(i_1, \ldots, i_n|i_j)$$

is well-defined for all move-forward rules, as we will demonstrate.
Let R be a move-forward rule defined by $1 = m_1 \leq \cdots \leq m_n$ with $m_i < i$ for $i \geq 2$. Let $W(k) = \{l \mid m_l = k\}$ be the set of all positions whose elements are

moved to position k upon request. The steady-state equations for the extended Markov chain are

$$q(i_1,\ldots,i_n|i_j) = \sum_{k=1}^{n} \sum_{l\in W(k)} p_{i_k i_j} \cdot q(i_1,\ldots,i_{k-1},i_{k+1},\ldots,i_l,i_k,i_{l+1},\ldots,i_n|i_k)$$

with the convention $q(i_1,\ldots,i_{k-1},i_{k+1},\ldots,i_l,i_k,i_{l+1},\ldots,i_n|i_k) = q(i_1,\ldots,i_n|i_1)$ in the case $k = l = 1$.
Using the above reduction and noting $\sum_{l=1}^{n} q(l) = 1$, we obtain

$$q(j) = \beta + (\alpha - \beta) \sum_{l\in W(j)} q(l) \qquad \text{for } j = 1,\ldots,n \tag{6}$$

This system of equations is consistent and possesses a unique solution. The $n \times n$ transition matrix P of the extended but reduced chain looks as follows:

$$P = \begin{pmatrix} \alpha & \beta & \beta & \beta & \cdots & \beta \\ \vdots & \beta & \beta & \beta & \cdots & \beta \\ \alpha & \beta & \beta & \beta & \cdots & \beta \\ \beta & \alpha & \beta & \beta & \cdots & \beta \\ \beta & \vdots & \beta & \beta & \cdots & \beta \\ \beta & \alpha & \beta & \beta & \cdots & \beta \\ & & \vdots & & & \\ \beta & \cdots & \beta & \alpha & \beta & \cdots \end{pmatrix} \tag{7}$$

In column k, exactly the elements p_{ik} with $i \in W(k)$ are α, all others are β.

Since $q(j)$ is the probability that the element on position j will be requested next, the expected search time is $\mu = \sum_{j=1}^{n} jq(j)$.

Theorem 7. *Given two move-forward rules R, R', their expected search times under the access sequence (5) and $\alpha \geq \beta$ obey*

$$R \leq R' \implies \mu(R') \leq \mu(R) \ .$$

For the proof we need the concepts of vector dominance and monotone matrices (see Keilson/Kester [9] for the background, Lam [12] for the application to independent accesses).
A probability vector $q = (q(1),\ldots,q(n))$ dominates another probability vector $q' = (q'(1),\ldots,q'(n))$, denoted by $q' \prec q$, if

$$\sum_{j=i}^{n} q'(j) \leq \sum_{j=i}^{n} q(j) \qquad \text{for all } i = 1,\ldots,n \ .$$

A Markov chain with transition matrix $P = (p_{ij})$ is stochastically monotone if for all fixed s, the partial sum of the ith row, $P_{is} = \sum_{j=1}^{s} p_{ij}$, is monotone non-increasing in i.

Lemma 8. *Let P and P' be two ergodic and stochastically monotone Markov chains with stationary probabilities q and q' respectively. Then*

$$qP' \prec q \implies q' \prec q$$

(see [9, 12]).

Now given two move-forward rules $R \leq R'$ under the access sequence (5), we show that in the situation $\alpha \geq \beta$, their transition matrices P, P' given by (7) are stochastically monotone and their stationary probabilities obey $qP' \prec q$. With the lemma, we get $q' \prec q$: $\sum_{j=i}^{n} q'(j) \leq \sum_{j=i}^{n} q(j)$ for all i. Summing over i yields $\sum_{i=1}^{n} iq'(i) \leq \sum_{i=1}^{n} iq(i)$, which is $\mu(R') \leq \mu(R)$.

From the definition of P it is easy to see that P is stochastically monotone in the case $\alpha \geq \beta$.

Since q is the stationary distribution for P, $q = qP$, and $qP' \prec q$ is equivalent to $qP' \prec qP$. Let

$$V(i) \;=\; \bigcup_{j=i}^{n} W(j) \;=\; \{l \mid m_l \geq i\} \;.$$

We demonstrate $qP' \prec qP$ in the situation $\alpha \geq \beta$. For all $i = 1, \ldots, n$, the following is equivalent

$$\begin{aligned}
\sum_{j=i}^{n} (qP')_j &\leq \sum_{j=i}^{n} (qP)_j \\
\sum_{j=i}^{n} \Big(\beta + (\alpha - \beta) \sum_{l \in W'(j)} q(l)\Big) &\leq \sum_{j=i}^{n} \Big(\beta + (\alpha - \beta) \sum_{l \in W(j)} q(l)\Big) \\
\sum_{j=i}^{n} \sum_{l \in W'(j)} q(l) &\leq \sum_{j=i}^{n} \sum_{l \in W(j)} q(l) \\
\sum_{l \in V'(i)} q(l) &\leq \sum_{l \in V(i)} q(l)
\end{aligned}$$

which is true since $m'_l \leq m_l$ for all $l = 1, \ldots, n$ entails $V'(i) \subseteq V(i)$ for all $i = 1, \ldots, n$. This completes the proof. □

4 Self-organizing binary search trees

In this section we briefly consider self-organizing search trees with the move-to-root heuristic. For independent accesses that are distributed with the distribution $(p_1, \ldots, p_n)$, the move-to-root heuristic achieves the expected search time

$$\mu \;=\; 1 + \sum_{i<j} \frac{p_i p_j}{p_i + \cdots + p_j} \;\leq\; 1 + 2 \ln 2 \cdot H(p)$$

where $H(p) = -\sum_i p_i \log_2 p_i$ is the entropy of the access distribution (see [1] or [6]).

For dependent accesses, we want to apply backward analysis. Hence we reformulate the lemma given in (Allen/Munro [1]).

Lemma 9. *Let $i < j$. When k is requested, j is an ancestor of i in the tree, if we trace back the sequence of accesses starting from k and encounter j before we encounter any of the elements $i, \ldots, j-1$. Similarly, when k is requested, i is an ancestor of j, if we trace back the access sequence starting from k and encounter i before we encounter any of the elements $i+1, \ldots, j$.*

In the case of move-to-front, we had to deal with singleton taboo sets. Now the taboo sets $H = \{i, \ldots, j-1\}$ and $H = \{i+1, \ldots, j\}$ respectively are of size $|j-i|$.
The probability $d(j, i|i)$ that j is an ancestor of i in the tree when i is accessed is given by

$$d(j,i|i) \;=\; {}_H\hat{f}^*_{ij} \;=\; \frac{\det((\delta_{ab}-1)\hat{m}_{ab} + \hat{m}_{ib} + \hat{m}_{aj} - \hat{m}_{ij})_{a,b\in H}}{\det((\delta_{ab}-1)\hat{m}_{ab} + \hat{m}_{jb} + \hat{m}_{aj})_{a,b\in H}}$$

with $H = \{i, \ldots, j-1\}$ and $\hat{m}_{ij}$ being the mean first passage time in the time-reversed chain of accesses (for $d(i, j|j)$ similarly), see (Kemperman [11]). The expected search time is $\mu = 1 + \sum_{j\neq i} q_i \cdot d(j, i|i)$. Unfortunately we did not succeed in deriving a nice expression giving μ in terms of the Markov chain of accesses.

We restrict our attention to a special sequence of requests. Given a probability distribution $q = (q_1, \ldots, q_n)$, all $q_i > 0$, we define

$$P = \lambda \cdot Q + (1-\lambda) \cdot I \tag{8}$$

where Q is the matrix of independent transitions with distribution q, I is the identity matrix, and $0 < \lambda \leq 1$. The stationary distribution of the ergodic chain P is q.

Theorem 10. *Let $\mu(Q)$ be the expected search cost for independent accesses with distribution q. The expected search cost under the move-to-root heuristic for the access sequence P defined by (8) is*

$$\mu(P) = \lambda \cdot \mu(Q) + (1-\lambda) \;. \tag{9}$$

This is exactly the same formula that was found for linear lists and the move-to-front rule under that access sequence (Lam/Leung/Siu [13]). See [15] for the derivation.

As a corollary, for all distributions q with $q_i > 0$ and all $\varepsilon > 0$ we can define a Markov chain with stationary distribution q such that the expected search cost under the move-to-root heuristic is $\mu(P) < 1 + \varepsilon$, by choosing λ sufficiently small, e.g. $\lambda = \min(1, \varepsilon/(\mu(Q) - 1))$.

Theorem 11. *For the Markov chain P defined by (8), the expected search cost is bounded by*

$$\begin{aligned}
\mu(P) &\leq 1 + 2\ln 2 \cdot \Big(H(P) + 1 - H(\lambda, 1-\lambda)\Big) \\
&\leq 1 + 2\ln 2 \cdot \Big(H(P) + 1\Big) \\
&\leq 3 + 2\ln 2 \cdot \Big(\mu_{\text{opt}} + \log \mu_{\text{opt}}\Big)
\end{aligned}$$

where $H(P)$ is the entropy of the chain, $H(.,.)$ is the entropy function and μ_{opt} is the asymptotically optimal search time which can be realised upon knowledge of P (see Hotz [7]).

The factor $2 \ln 2$ is the same as for independent accesses. This is interesting since neither $H(P)$ nor μ_{opt} reflect the locality of accesses. It would be desirable to establish a bound on the expected search cost of move-to-root for a broader range of access sequences.

References

1. B. Allen and I. Munro. Self-organizing binary search trees. *Journal of the ACM*, 25(4):526–535, 1978.
2. P. J. Burville and J. F. C. Kingman. On a model for storage and search. *Journal of Applied Probability*, 10:697–701, 1973.
3. Ph. Chassaing. Optimality of move-to-front for self-organizing data structures with locality of references. *Annals of Applied Probability*, 3(4):1219–1240, 1993.
4. F. R. K. Chung, D. J. Hajela, and P. D. Seymour. Self-organizing sequential search and Hilbert's inequalities. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 217–223, New York, 1985.
5. G. H. Gonnet, J. I. Munro, and H. Suwanda. Exegesis of self-organizing linear search. *SIAM Journal of Computing*, 10(3):613–637, 1981.
6. G. H. Gonnet, and R. Baeza-Yates. *Handbook of Algorithms and Data Structures.* Addison-Wesley, 1991.
7. G. Hotz. Search tress and search graphs for markov sources. *Journal of Information Processing and Cybernetics*, 29:283–292, 1993.
8. Y. C. Kan and S. M. Ross. Optimal list order under partial memory constraints. *Journal of Applied Probability*, 17:1004–1015, 1980.
9. J. Keilson and A. Kester. Monotone matrices and monotone markov processes. *Stochastic Processes and Applications*, 5:231–241, 1977.
10. J. G. Kemeny and J. L. Snell. *Finite Markov Chains.* Van Nostrand, Princeton, 1960.
11. J. H. B. Kemperman. *The First Passage Problem for a Stationary Markov Chain.* University Press, Chicago, 1961.
12. K. Lam. Comparison of self-organizing linear search. *Journal of Applied Probability*, 21:763–776, 1984.
13. K. Lam, M. Y. Leung, and M. K. Siu. Self-organizing files with dependent accesses. *Journal of Applied Probability*, 21:343–359, 1984.
14. R. I. Phelps and L. C. Thomas. On optimal performance in self-organizing paging algorithms. *Journal Inf. Optimization Science*, 1:80–93, 1980.
15. F. Schulz. Self-organizing data structures with dependent accesses. MSc Thesis supervised by Prof. Dr. G. Hotz, University of Saarbrücken, 1995 (in German). See http://hamster.cs.uni-sb.de/~schulz/
16. A. M. Tenenbaum and R. M. Nemes. Two spectra of self-organizing sequential search algorithms. *SIAM Journal of Computing*, 11:557–566, 1982.

This article was processed using the LaTeX macro package with LLNCS style