A TIGHT AMORTIZED BOUND FOR PATH REVERSAL

David Ginat
Daniel D. Sleator
Robert E. Tarjan

CS-TK-153-88

June 1988

October 6, 1988
A Tight Amortized Bound for Path Reversal

David Ginat

Daniel D. Sleator

Robert E. Tarjan

June, 1988

ABSTRACT

Path reversal is a form of path compression used in a disjoint set union algorithm and a mutual exclusion algorithm. We derive a tight upper bound on the amortized cost of path reversal.
A Tight Amortized Bound for Path Reversal

David Ginat

Daniel D. Sleator

Robert E. Tarjan

June, 1988

Let $T$ be a rooted $n$-node tree. A path reversal at a node $x$ in $T$ is performed by traversing the path from $x$ to the tree root $r$ and making $x$ the parent of each node on the path other than $x$. Thus $x$ becomes the new tree root. (See Figure 1.) The cost of the reversal is the number of edges on the path reversed. Path reversal is a variant of the standard path compression algorithm for maintaining disjoint sets under union [5]. It has also been used in a novel mutual execution algorithm [2,6].

Figure 1. Path reversal. Triangles denote subtrees.

---

1 Department of Computer Science, University of Maryland.
2 Department of Computer Science, Carnegie-Mellon University. Research partially supported by DARPA, ARPA order 4976, amendment 19, monitored by the Air Force Aeronautics Laboratory under Contract No. F33615-87-C-1499, by the National Science Foundation under Grant No. CCR 8658139, and by AT&T Bell Laboratories.
3 Department of Computer Science, Princeton University, and AT&T Bell Laboratories. Research partially supported by NSF Grant No. DCR-8605962 and ONR Contract No. N00014-87-K-0467.
Suppose that a sequence of \( m \) reversals is performed on an arbitrary initial tree. What is the total cost of the sequence? Let \( T(n,m) \) be the worst-case cost of such a sequence, and let \( A(n,m) = T(n,m)/m \). We are most interested in the value of \( A(n,m) \) for fixed \( n \) as \( m \) grows. As discussed by Tarjan and Van Leuwen [5], binomial trees provide a class of examples showing that \( A(n,m) \geq \lceil \log n \rceil \), and their rather complicated analysis gives an upper bound of \( A(n,m) = O(\log n + \frac{n \log n}{m}) \). Gmit and Shankar [2] prove that \( A(n,m) \leq 2 \log n + \frac{n \log n}{2m} \). We shall prove that \( A(n,m) \leq \log n + \frac{n \log n}{2m} \). In the special case that the initial tree consists of a root with \( n-1 \) children, which is the case in the mutual exclusion algorithm, the bound is \( A(n,m) \leq \log n \).

To obtain the bound, we apply the potential function method of amortized analysis. (See [4].) Let the size \( s(x) \) of a node \( x \) in \( T \) be the number of descendants of \( x \), including \( x \) itself. Let the potential of \( T \) be \( \Phi(T) = \frac{1}{2} \sum_{x \in T} \log s(x) \). Define the amortized cost of a path reversal over a path of \( k \) edges to be \( k - \Phi(T) + \Phi(T') \), where \( T \) and \( T' \) are the trees before and after the reversal, respectively. For any sequence of \( m \) reversals, we have

\[
\sum_{i=1}^{m} a_i = \sum_{i=1}^{m} (t_i - \Phi_{i-1} + \Phi_i) = \sum_{i=1}^{m} t_i - \Phi_0 + \Phi_m,
\]

where \( a_i \), \( t_i \), and \( \Phi_i \) are the amortized cost of the \( i^{th} \) reversal, the actual cost of the \( i^{th} \) reversal, and the potential after the \( i^{th} \) reversal, respectively, and \( \Phi_0 \) is the potential of the initial tree. Since \( \Phi_0 \leq \frac{n}{2} \log n \) and \( \Phi_m \geq \frac{1}{2} \log n \), this inequality yields

\[
\sum_{i=1}^{m} t_i \leq \sum_{i=1}^{m} a_i + \frac{1}{2} (n-1) \log n,
\]

which in turn implies

\[
A(n,m) \leq \frac{1}{m} \sum_{i=1}^{m} a_i + \frac{n \log n}{2m}.
\]

* All logarithms in this paper are base two.
We shall prove that the amortized cost of any reversal is at most \( \log n \), thereby showing that

\[
A(n, m) \leq \log n + \frac{n \log n}{2m}.
\]

When the initial tree consists of a root with \( n-1 \) children, the bound drops to

\[
A(n, m) \leq \log n,
\]

since then \( \Phi_0 \leq \Phi_m \), and the extra additive term drops out.

Let \( x_0, x_1, x_2, \ldots, x_k \) be a path that is reversed, and let \( A \) be the amortized cost of the reversal. For \( 0 \leq i \leq k \), let \( s_i \) be the size of \( x_i \) before the reversal. The size of \( x_0 \) after the reversal is \( s_k \), and the size of \( s_i \) after the reversal, for \( 1 \leq i \leq k \), is \( s_i - s_{i-1} \). We can thus write \( A \) as

\[
A = k - \sum_{i=0}^{k-1} \frac{1}{2} \log s_i + \frac{1}{2} \log s_k + \sum_{i=1}^{k} \frac{1}{2} \log (s_i - s_{i-1})
\]

\[
= k + \frac{1}{2} \sum_{i=0}^{k-1} \left( \log (s_{i+1} - s_i) - \log s_i \right)
\]

\[
= k + \frac{1}{2} \sum_{i=0}^{k-1} \log \left( \frac{(s_{i+1} - s_i)}{s_i} \right).
\]

For \( 0 \leq i \leq k-1 \), let \( \alpha_i = \frac{s_{i+1}}{s_i} \). Note that \( \frac{(s_{i+1} - s_i)}{s_i} = \alpha_i - 1 \). We have

\[
A = k + \frac{1}{2} \sum_{i=0}^{k-1} \log (\alpha_i - 1)
\]

\[
= \sum_{i=0}^{k-1} \left( 1 + \frac{1}{2} \log (\alpha_i - 1) \right)
\]

We now make use of the following inequality, which will be verified below: for all \( \alpha > 1 \),

\[
1 + \frac{1}{2} \log(\alpha - 1) \leq \log \alpha.
\]

From this inequality we obtain

\[
A \leq \sum_{i=0}^{k-1} \log \alpha_i
\]

\[
= \sum_{i=0}^{k-1} \log \left( \frac{s_{i+1}}{s_i} \right) = \sum_{i=0}^{k-1} \left( \log s_{i+1} - \log s_i \right)
\]

\[
= \log s_k - \log s_0
\]

\[
\leq \log n,
\]

since \( s_k = n \) and \( s_0 \geq 1 \).
This completes the amortized analysis. We verify the needed inequality by the following chain of reasoning:

\[ 0 \leq (\alpha-1)^2 \]
\[ \Rightarrow 0 \leq \alpha^2 - 4\alpha + 4 \]
\[ \Rightarrow 4 (\alpha - 1) \leq \alpha^2 \]
\[ \Rightarrow \log (4(\alpha - 1)) \leq \log (\alpha^2) \]
\[ \Rightarrow 2 + \log (\alpha - 1) \leq 2\log \alpha \]
\[ \Rightarrow 1 + \frac{1}{2} \log (\alpha - 1) \leq \log \alpha. \]

We conclude with some remarks. The definition of the potential function used here has been borrowed from Sleator and Tarjan's analysis of splay trees [3]; it has also been used to analyze pairing heaps [1]. As in the case of splay trees, the upper bound can be generalized in the following way. Assign to each tree node \( x \) a fixed but arbitrary positive weight \( w(x) \). Define the total weight of \( x \), \( tw(x) \), to be the sum of the weights of all descendants of \( x \), including \( x \) itself. Define the potential of the tree \( T \) to be \( \Phi(T) = \frac{1}{2} \sum_{x \in T} \log tw(x) \). A straightforward extension of the above analysis shows that the total cost of a sequence of \( m \) reversals is at most \( \sum_{i=1}^{m} \log \left(\frac{W/w_i}{w_i}\right) + \Phi_0 - \Phi_m \), where \( w_i \) is the weight of the node \( x_i \) at which the \( i \)th reversal starts and \( W \) is the sum of all the node weights.

Choosing \( w(x) = 1 \) for all \( x \in T \) gives our original result. Choosing \( w(x) = f(x) + 1 \), where \( f(x) \) is the number of times a reversal begins at \( x \), gives an upper bound for the total time of all reversals of \( \sum_{i=1}^{m} \log \left(\frac{n+m}{f(x_i)}\right) + \frac{1}{2} \sum_{x \in T} \log \left(\frac{n+m}{f(x)}\right) \).

It is striking that the "sum of logarithms" potential function serves to analyze three different data structures. We are at a loss to explain this phenomenon; whereas there is a clear connection between splay trees and pairing heaps (see [1]), no such connection between trees with path reversal and the other two data structures is apparent. In the case of path reversal, the sum of logarithms potential function gives a bound that is exact to within an additive term depending only on the initial and final trees. It would be extremely interesting and useful to have a systematic method for deriving appropriate potential functions. The three examples of splaying, pairing, and reversal offer a setting in which to search for such a method.
Acknowledgement. The first author thanks D. Mount and A. U. Shankar for valuable discussions and useful comments.

References


