Analysis of height

Suppose the B-tree has height $h$

What is the min # of elements it might hold?

$n \geq f(h)$ solve for $h \leq f(n)$
torder

h

# elements

level 0

1

2 \( (t-1) \)

2t \cdot (t-1)

\frac{\# \text{nodes}}{2t^2(t-1)}

\frac{\# \text{nodes}}{2t^3(t-1)}

level \ h-1

2t^{h-1}(t-1)
\[ n \geq 1 + 2(t-1) \left( 1 + t + t^2 + \cdots + t^{h-1} \right) \]

\[ n \geq 1 + 2(t-1) \left( \frac{t^h - 1}{t-1} \right) \]

\[ n \geq 1 + 2t^h - 2 \]

\[ n \geq 2t^h - 1 \]

\[ 2t^h \leq n + 1 \]

\[ \sum_{i=0}^{x} r_i = \frac{x+1}{x+1} - 1 \]
\[ 2t^h \leq n+1 \]
\[ t^h \leq \frac{n+1}{2} \]
\[ h \leq \log_t \left( \frac{n+1}{2} \right) \]

\[ \log_t \left( \frac{n+1}{2} \right) \]
\[ = \log_t (n+1) - \log_t 2 \]
\[ \approx \log_t n \]

Maximum height of B-tree with \( n \) elements.
Cost for insertion

\[ \Theta \left( \log_t n \cdot \log_2 (2t-1) \right) = \Theta(\log_2 n) \]

- # nodes on path down
- time per node (sorted array)
- Max # of page faults
Overview of B-tree Deletion

Like binary search tree, for removing element in an internal node, then replace $x$ by its successor and remove the successor from marked node. Take leftmost child until reach leaf & successor. Leftmost element.