Cost Expression for Index Scan

Amit Gupta (amit_gupta@persistent.co.in, amit.pc.gupta@gmail.com)

Some parameters:

- h: Height of the B-tree
- f: Avg fan-out of the B-tree nodes.
- x: Number of key searches (probes) on the B-tree
- b: Size of buffer cache in terms of number of blocks
- n: Number of probes that fill the cache. Cache reaches steady state after n probes.

Problem Definition: Compute the expected number of index block IO's required to execute x probes on a B-tree index.
Probability that an index node at i^{th} level (of the B-tree) is not found in cache during x^{th} index probe is described in the table below.

	B-tree level=1	**B-tree level=2**	...	**h**
1^{st} **probe**	1	1	...	1
2^{nd} **probe**	0	$1-1/f$	...	$1-1/f^{h-1}$
2^{rd} **probe**	0	$(1-1/f)^2$	...	$(1-1/f^{h-1})^2$
...	.	.	.	.
x^{th} **probe**	0	$(1-1/f)^{x-1}$	...	$(1-1/f^{h-1})^{x-1}$
Total IO =	1	$\sum_{r=0}^{x-1}(1-1/f)^r$	...	$\sum_{r=0}^{x-1}(1-1/f^{h-1})^{x-1}$

- Assuming that the index probe starts with empty cache, for the first index probe all the B-tree nodes from root to leaf will be read from disk. This represented in the table above as 1 block read for each B-tree level.
- The second probe will find root node in cache, but the probability of finding second level index node in cache is 1/f, since there are f second level nodes. Similarly, as only 1 leaf node is in cache, probability of finding it out of f^{h-1} blocks is $1/f^{h-1}$.
- In the third probe, probability of not finding the desired node at second level is $(1-1/f)^2$.

Total block IO for x index probes =

$$1+\sum_{r=0}^{x-1}(1-1/f)^r+\sum_{r=0}^{x-1}(1-1/f^2)^r+\ldots+\sum_{r=0}^{x-1}(1-1/f^{h-1})^{x-1}$$

By applying sum of geometric expressions, we get the following result.

$$=1+\frac{1-(1-1/f)^x}{1/f}+\ldots+\frac{1-(1-1/f^{h-1})^x}{1/f^{h-1}}$$

Using Binomial theorem,

$$=1+f(\frac{^xC_1}{f}-\frac{^xC_2}{f^2}+\ldots+\frac{^xC_k}{f^k}+\ldots)+f^2(\frac{^xC_1}{f^2}-\frac{^xC_2}{f^2}+\ldots+\frac{^xC_k}{f^{2k}}+\ldots)+$$

$$\ldots+f^r(\frac{^xC_1}{f^r}-\frac{^xC_2}{f^{2r}}+\ldots+\frac{^xC_k}{f^{rk}}+\ldots)+\ldots$$

$$=1+x(h-1)-\frac{^xC_2}{f}(1+\frac{1}{f}+\ldots+\frac{1}{f^{h-2}})+\frac{^xC_3}{f^2}(1+\frac{1}{f^2}+\ldots+\frac{1}{f^{2(h-2)}})+$$

$$\ldots+\frac{^xC_k}{f^{k-1}}(1+\frac{1}{f^{k-1}}+\ldots+\frac{1}{f^{(k-2)(h-2)}})+\ldots$$

Applying the sum of geometric series.

$$=1+x(h-1)-\frac{^xC_2}{f}\times\frac{1-1/f^{h-1}}{1-1/f}+\frac{^xC_3}{f^2}\times\frac{1-1/f^{2(h-1)}}{1-1/f^2}\ldots+\frac{^xC_k}{f^{k-1}}\times\frac{1-1/f^{(k-1)(h-1)}}{1-1/f^{k-1}}+\ldots$$

Assuming that $f >> 1, \implies 1/f \rightarrow 0$

$$= 1 + x(h-1) - \frac{{}^{x}C_2}{f} + \frac{{}^{x}C_3}{f^2} + \ldots + \frac{{}^{x}C_k}{f^{k-1}} + \ldots$$

$$= 1 + x(h-1) + \frac{1}{f} - \frac{x}{f^2} - \frac{(1-1/f)^x}{f}$$

$$= 1 + x(h-1) + \frac{1}{f} - \frac{x}{f^2} - \frac{(1-1/f)^x}{f}$$

$$= 1 + x(h-1) - \frac{x}{f^2}, \quad when \quad 1/f \rightarrow 0$$

When cache reaches steady state $(x = n)$

$$1 + n(h-1) - n/f^2 = b \implies n = \frac{b-1}{h-1}$$

$$Num \quad IO's = 1 + x(h-1) + \frac{x}{f^2}, if \quad x <= n$$

When $x > n$, let first l levels of the B-tree are cached. $l = \lfloor log_f b \rfloor$
The remaining cache $R = b - f^l$ can be utilized caching other B-tree nodes (from $l+1$ to h levels).

$$Num \quad IO = b + (x-n)(h-l)\frac{f^{l+1} + \ldots + f^{h-1}}{R}, When \quad x > n$$

$$= b + (x-n)(h-l)\frac{f^{h-1} - f^l}{R}, assuming \quad f >> 1$$

References

1. Index scans using a finite LRU buffer: a validated I/O model, Lothar F. Mackert, Guy M. Lohman