Shadow Storage

Idea: keep two versions for each *modified* block:

old, consistent version

new, (possibly) inconsistent version

atomic switch to indicate consistent versions

**but: two mapping tables**
Example: Modifying Transaction T1

version a

file

\( M_a \)

\[ \begin{array}{cccccc}
B0 & B1 & B2 & B3 & B4 & B5 \\
\end{array} \]

\[ \begin{array}{cccccc}
B6 & B7 & B8 & B9 & a \\
\end{array} \]

pointer to consistent version
Insert and Update

version a

file

version b

pointer to consistent version
Insert and Update

version a

file

version b

pointer to consistent version
Crash

version a

file

version b

$M_a \begin{array}{c} 0 \ 1 \ 2 \ 3 \ 4 \ 5 \end{array}$

$M_b \begin{array}{c} 0 \ 1 \ 2 \ 3 \ 4 \ 5 \end{array}$

pointer to consistent version
Persisting Changes

version a

file

version b

1. Write modified blocks
2. Write M_b
3. Perform global switch

pointer to consistent version
Discussion

Advantages:

Doubles storage only for changed blocks
undo of changes easy

Disadvantages:

helper data structures (maps) may become "big" (> 1 block)
high degree of fragmentation

virtual memory
ZFS