veribetrfs

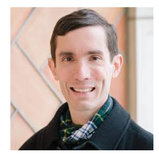
a faster filesystem that can't lose data
a software development methodology beyond testing


Travis Hance


Jon Howell

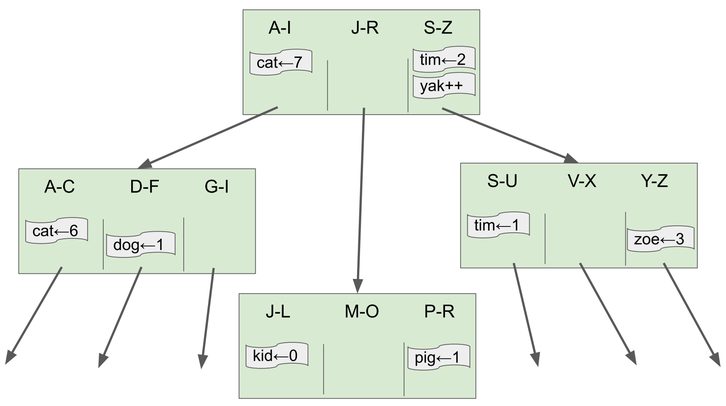

Rob Johnson


Andrea Lattuada

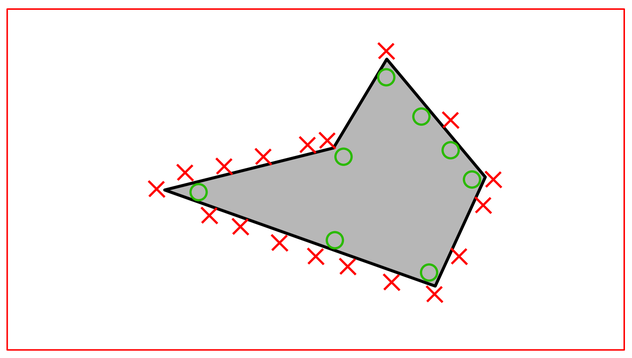

Bryan Parno

B$^\varepsilon$tree

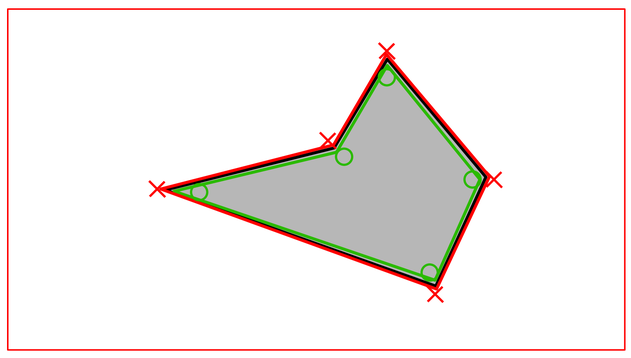
messages enter at the top of the tree
enabling coalesced writes
trade write amplification against slow seeks
for fast random writes and fast sequential reads



Test-driven development checks a
pointwise approximation specification



Verification-driven development checks an
exhaustive specification



write a concise, abstract spec

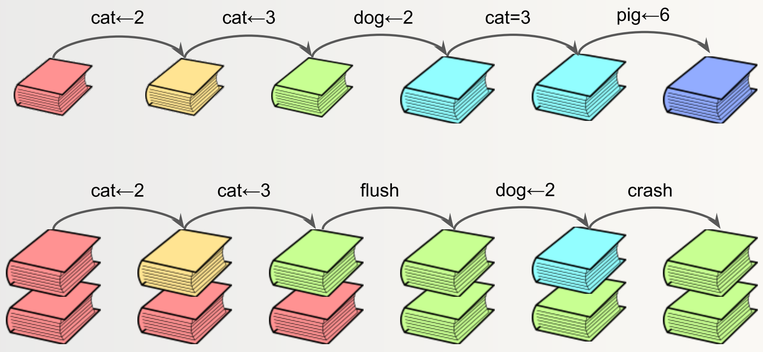
An abstract spec for the Shortest Path function

```
predicate IsPath(g, p, begin, end) {
  && p[0] == begin
  && Last(p) == end
  && forall i :: 0 <= i < |p|-1
       ==> Edge(p[i], p[i+1]) in g.edges
}

predicate IsShortestPath(g, p, begin, end) {
  && IsPath(g, p, begin, end)
  && forall p2 :: IsPath(g, p2, begin, end)
       ==> Len(p) <= Len(p2)
}
```

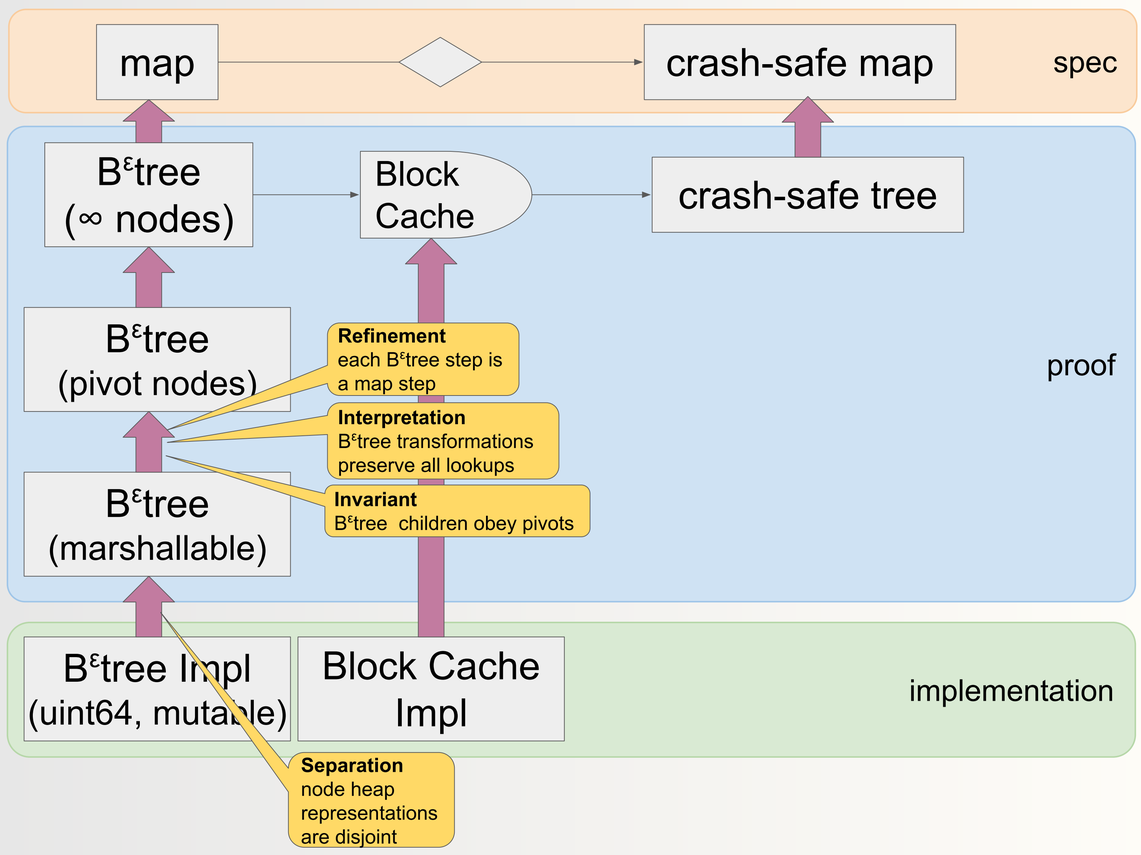
A state-machine spec for a KV store

```
datatype Variables = Variables(view:map<K,V>)

predicate Query(key:Key, result:Value) {
  && result == s.view[key]
  && s' == s
}
predicate Write(key:Key, new_value:Value) {
  && s'.view == s.view[key := new_value]
}

predicate Init() {
  s.view == EmptyMap()
}
predicate Next() {
  || (exists k, r :: Query(k, r))
  || (exists k, v :: Write(k, v))
}
```

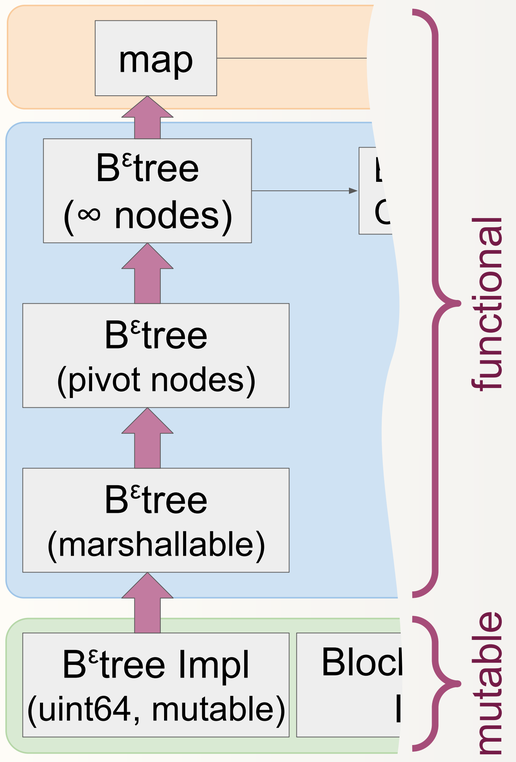
A state-machine spec for a persistent KV store



organize a maintainable proof artifact



exploit automation & cull timeouts




```
lemma NodeIsWellFormed(c:Children, n: Node)
  requires n == ConstructNode(c)
  ensures WellFormed(n)
{
  assert Sorted(n.children);
}
```

contain aliasing reasoning

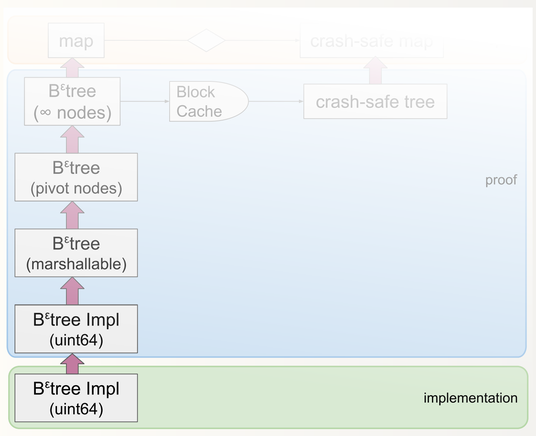
Aliasing confounds functional reasoning

```
method foo() {
  n1 := MakeNode(parent);
  i := this.FindIndex(key);
  this.UpdateWeights();
  this.ReplaceChild(i, n1);
}
```

ObeysPivots(n1)

IndexForKey(key, i)

Iron★ Model-Impl mates restore bisection debugging



Rust borrow types might help!