Rethinking Composition

Michael Spear

(Written in the last 5 minutes, so please be extra cruel 😊)
Transactions with Isolation and Cooperation
Yannis Smaragdakis et al.
OOPSLA 2007
The “transactional barrier”

Initially:    const int nthreads = 16;
              int counter = 0;

atomic { counter ++ }
atomic { while (counter != nthreads) retry }
Why Not?

atomic {
    atomic { counter ++ }
    atomic {while (counter != nthreads) retry }
}

If counter == 0; I can never increment it and see others increment it, so it can’t ever reach nthreads if nthreads > 1!
From M. Scott: This is the same

```java
atomic {while (counter != nthreads) retry } ==

bool b = true
while (b) {
    atomic {
        atomic {
            b = (nthreads > counter)
        }
    }
}
```
Key Point:

• This has nothing to do with retry

• It has everything to do with composition not being what we think
  – Seems that composability depends on transitive data dependencies and control flow
Next Point: Semantics vs. Nesting

• Consider ALA semantics (tl2+IP)
• If a nested transaction aborts, the outer transaction must abort
  – Nested transaction reads some datum that changed after it started
  – Child timestamp must == parent timestamp
  – Can’t change parent timestamp without aborting it
• Note: NOrec can avoid this… no other ALA system can (yet?)
Can’t prefetch data, go irrevocable, and then validate the data

initially val = 42, flag = false, r = 0

T1:
atomic {
    x = val

    become irrevocable
    if (flag)
        r = x;
}
can r == 42?

T2:
val = 0
atomic { flag = true }
Summary

• Can someone please formalize composition?
• Can you please include irrevocability and language-level semantics?