On The Orthogonality of Speculation and Atomicity

Michael L. Scott
Luke Dalessandro

Department of Computer Science
University of Rochester

WTTM2, Boston, MA
September 16, 2010
Is TM Speculative?

- TM developed to allow composable atomicity for non-blocking data structure
- We (and others) have developed TM semantics which provide only atomicity—but TM implementations are typically speculative.
- Temptation to expose speculation and provide operations like...
abort (orElse)

- Allows the programmer to abandon a transaction when a precondition does not hold
- Requires an explicit notion of speculation because it leaks the fact that precondition does not hold—complicates our simple model
- If you really want them we can optionally add such speculation
Claims

- It is possible to integrate the notion of explicit speculation into a memory model directly, independently of atomicity.
- Composing such a model with atomicity naturally allows us to model traditional mechanisms like *abort*. 
Main Idea: `spec {}` block

```plaintext
spec {
    ...
    if (!C)
        fail
    ...
}

spec {
    ...
    if (!C)
        fail
    ...
}
else {
    // syntactic sugar
}

cf. Shinnar et al. [MSR-TR-2004]
```
Formalizing Speculation

- Ordering-based: extend thread histories with speculation markers and introduce orders and rules that “hide” failed writes from future reads
- Syntax-based: expand *fail* in terms of compensating writes and use traditional ordering-based rules
- Semantics differ for parallel execution
Sequential Specification

visibility order

Good enough for “sequential” contexts

fail operation “hides” writes
Modeling \textit{abort}

txn {  
  ...
  if (!C)
  abort
  ...
}

atomic {  
  spec {  
    ...
    if (!C)
    fail
    ...
  }
}
Modeling `orElse`
Open Questions

• parallel specification of \texttt{spec} \{ \}
• should speculation be contagious?
• is \textit{atomic/spec} orthogonality desirable? (probably possible)
• nesting atomic in speculative regions
• relaxed models and racy programs
Final Thoughts

• We can add speculation cleanly
• spec-inside-atomic-idioms capture abort, orElse, more?
• Interesting future work remains