

Software Issues for Multicore Systems

Burton Smith
Microsoft

This subject is familiar

- In spite of clear signs in the mid-80's that a train wreck was coming, we let it happen
- What we failed to do then we must do now:
 - Pursue high-level parallel languages
 - Develop optimizing compilers for them
 - Provide architectural support for them

What happened?

- Transistor costs keep falling
- Chip suppliers and dependent businesses need roughly constant prices
- Instruction-level parallelism has run dry
 - Diminishing payoff for Herculean effort
 - We need multiple program counters (duh)
- It's not really due to power or clock limits

Libraries aren't the answer

- Our parallelism models don't compose
 - Is parallelism inside or outside the library?
 - Where is the synchronization done?
- Given the right language, libraries will help
 - To capture expertise
 - To leverage abstraction

We need old new things

- Dependence analysis
 - To guide parallelism optimization
- Loop restructuring
 - To optimize for the target system
- Functional programming
 - To avoid state whenever practical
- Transactions
 - To make state updates “commute”

Commutativity via transactions

- Operations on state seldom commute
 - Unlike pure expression evaluation
- Invariants are preserved, however
- Owicki/Gries: If the S_i are independent,

$$\frac{\{I\}S_1\{I\} \wedge \dots \wedge \{I\}S_n\{I\}}{\{I\}S_1 \parallel \dots \parallel S_n\{I\}}$$

- Transactions ensure independence
- We get commutativity modulo I