“If you can dodge a wrench, you can dodge a ball”

June 13, 2015

Dylan Stark
George Stelle
“It is important to understand what you CAN DO before you learn to measure how well you seem to have DONE IT.”

–John Tukey
“It is important to understand what you CAN DO before you learn to measure how well you seem to have DONE IT.”

–John Tukey

**START WITH THE BASICS:**

UNBALANCED TREE SEARCH (UTS)

INTEL XEON IVY BRIDGE (20-core dual-socket)
UTS on 20-core dual-socket Ivy Bridge

Harmonic mean of best rate (tasks/s)

Num. cores

defaults
- Chapel-flat
- Chapel-NUMA-m
- Chapel-NUMA-n
- Qthreads-m
- Qthreads-n
UTS on 20-core dual-socket Ivy Bridge

Percent diff. w.r.t. default Qthreads

Num. cores

defaults
- Chapel-flat
- Chapel-NUMA-m
- Chapel-NUMA-n
- Qthreads-m
- Qthreads-n
“It is important to understand what you CAN DO before you learn to measure how well you seem to have DONE IT.”

–John Tukey

**CHANGE THE APPLICATION (REQUIREMENTS):**

STREAM

INTEL XEON IVY BRIDGE (20-core dual-socket)
Stream on 20-core dual-socket Ivy Bridge

Add

Copy

Scale

Triad

Best rate (MB/s)

Num. cores

5 10 15 20

5 10 15 20
Stream on 20-core dual-socket Ivy Bridge

Add

Copy

Scale

Triad

Harmonic mean of best rate (MB/s)

Num. cores
Stream on 20-core dual-socket Ivy Bridge

- Add
- Copy
- Scale
- Triad

Harmonic mean of best rate (MB/s)

Num. cores

defaults
- Chapel-flat
- Chapel-NUMA-m
- Chapel-NUMA-n
- Qthreads-m
- Qthreads-n
“It is important to understand what you CAN DO before you learn to measure how well you seem to have DONE IT.”

–John Tukey

CHANGE THE ARCHITECTURE (CAPABILITIES):

STREAM

INTEL XEON PHI (57-core)
“It is important to understand what you CAN DO before you learn to measure how well you seem to have DONE IT.”

—John Tukey

**RINSE AND REPEAT:**

STREAM & UNBALANCED TREE SEARCH (UTS)

INTEL XEON HASWELL (32-core)
“It is important to understand what you CAN DO before you learn to measure how well you seem to have DONE IT.”

–John Tukey

Beware what you don’t know you don’t know about what the runtimes are doing (there lie extraneous variables, and confounders!)

But keep exploring with more applications and architectures
Any questions?