TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

TU Dresden, Center for Information Services and HPC (ZIH)

ALWAYS ON?

ENVISIONING FULLY-INTEGRATED PERMANENT MONITORING IN PARALLEL APPLICATIONS

Andreas Knüpfer

VI-HPS 10th Anniversary Workshop, Seeheim, 2017-06-23

Past Achievements: Score-P Community Software

Since 2007/2009 a group of VI-HPS partner institutions jointly develop and maintain the Score-P and OTF2 software packages for parallel runtime monitoring and recording

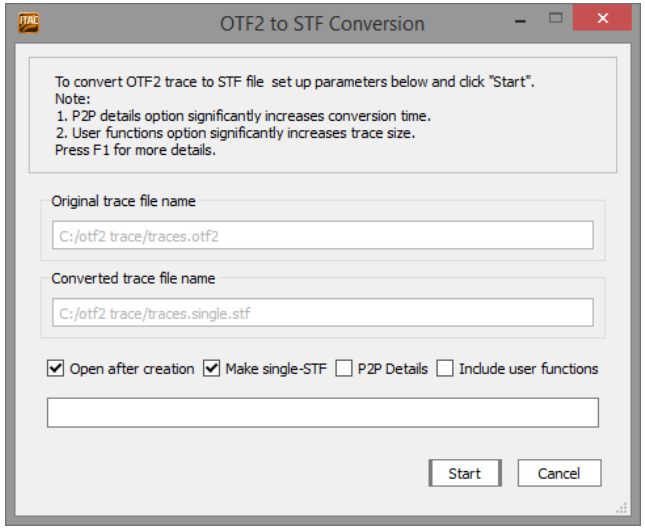
Score-P
Scalable performance measurement infrastructure for parallel codes

OTF2

- Many features
- Used worldwide

Thank you very much, dear partners!

Lately, the Intel Trace Analyzer supports OTF2 traces!



See https://software.intel.com/en-us/node/561577 and https://software.intel.com/en-us/node/684660 for the screenshot.

Who should tell about parallel performance?

Who offers parallel performance?


Various parallel programming models -- they give no insight into parallel performance


Hardware -- exposes some information but indirectly


Parallel libraries -- they tell nothing about performance


Applications -- some give a high-level performance report

Who tells about performance?

- Dedicated third-party tools
- Need to support exactly your combination of language, parallel model(s), and architecture(s)
- Few standards across tools
- Rather complicated to use

Without third-party tools one has **no clue** if it is running fast or efficient!

Let's compare to something slightly more usable …

From https://de.wikipedia.org/wiki/Datei:Mazda6_Typ_GJ_2.0_SKYACTIV-G_165_I-ELOOP_Sports-Line_Cockpit_Kombiinstrument_Tacho_beleuchtet_Nacht.jpg under Creative Commons license by Kickaffe (Mario von Berg)

- How fast is the car?
- How efficient is the car?
- Is something wrong?

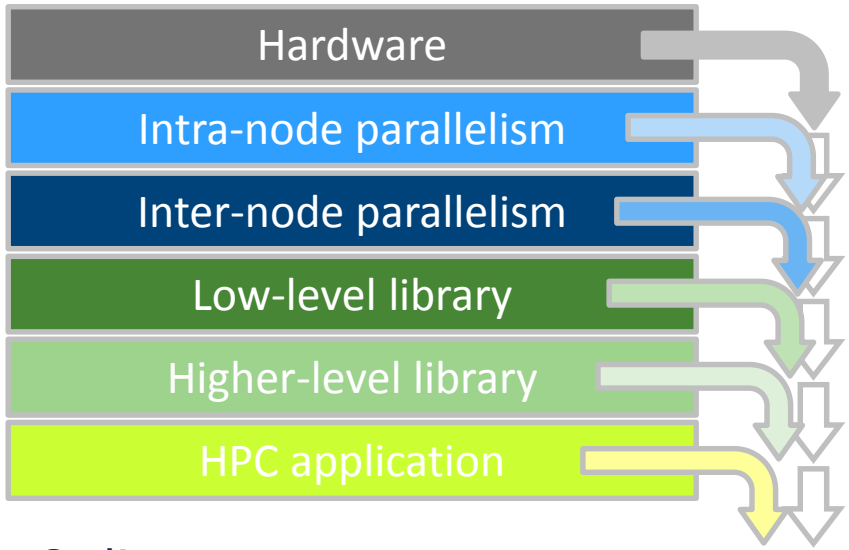
The usual "instrumentation" and displays in a car

- Speedometer, RPM meter, Odometer, Fuel mileage*
- Green, yellow, red indicator lights (a.k.a. "idiot lights")

- No special skills to read it
- No third-party tools needed
- Basically the same for Volkswagen, BMW, Ford, … (bear with non-SI units)
- Online, not post-mortem

Vision for the Future of Parallel Performance Monitoring

- Every layer comes with integrated performance reporting
- Define metrics relevant on the current level
- Pass on lower-level data

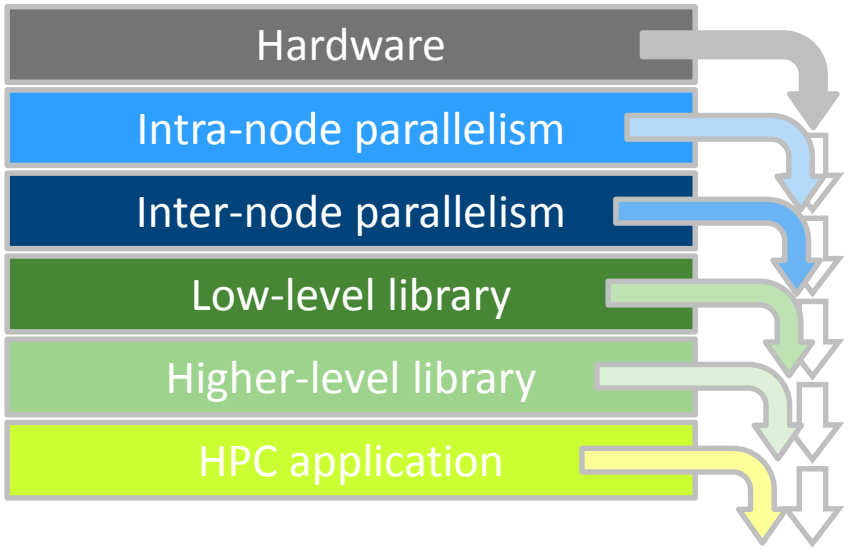


- Always on, no off switch!

https://commons.wikimedia.org/wiki/File:On-Off_Switch.jpg
Creative Commons Attribution-Share Alike 2.5 Generic license by Jason Zack at en.wikipedia

- Online
- Sensible overhead
- Little data by default, more on request
- Standardized APIs and formats

Example



CPUs offer HW counters*, GPUs offers CUPTI*

OpenMP: something based on OMPT*

MPI: something based on MPIT*

BLAS library: report vectors per second for certain vector lengths and operations

Multigrid library: report V-cycles per second

Application: report time to solution for so and so many degrees of freedom

* Selection required

- Always generate combined performance meters/report, not partially
- Provides conventional performance metrics plus new ones
- Allows to assess performance relationships
  - Flop/s go up but vector throughput stays constant => something is wrong

Exploring with the DASH PGAS library

dash
www.dash-project.org
Distributed Data Structures and Parallel Algorithms

- DASH is a high-level PGAS abstraction for parallel C++
  - Global data container classes with built-in data distribution information
  - Parallel template algorithms pay attention to distribution
  - Towards Exascale level
  - Slow global PGAS accesses vs. fast access to local parts
  - Slow individual remote access vs. fast bulk access
  - Implemented on top of MPI or GASPI

Is DASH being used efficiently?

- Introduce DASH-specific metrics about local and remote accesses
- Report results next to MPI/GASPI metrics, not intermixed or replacing them

(WP3.3 in *Smart DASH* project in the DFG SPPEXA program, 2017 - 2019)

How would this change the HPC tools landscape?

Possible advantages

- Performance always visible
  - Harder to ignore bad performance? ("idiot lights"?)
- Usability improves
  - Component maintainers do it, not external experts
  - It is always active, so issues show up earlier
- Interchangeable data formats?

Would "tools" as separate pieces of software go away?

- Maybe yes for the runtime part, but there will always be runtime infrastructure and special cases.
- Tools will stay around for sure for the analysis parts.
- New tools for analysis of higher-level components?

This should be the beginning of the discussion …

Disclaimer

- This is a personal opinion and vision how things should have been and should become
- Blame all crazy things and mistakes to the presenter
- All the hard work and the results and insights from it would have been impossible without the whole team at TU Dresden and many partners, of course!

Mario Bielert, Ronny Brendel, Holger Brunst, Robert Dietrich, Jens Doleschal, Ronald Geisler, Andreas Gocht, Christian Herold, (Tobias Hilbrich), Denis Hünich, Thomas Ilsche, (Guido Juckeland), Andreas Knüpfer, Matthias Lieber, Hartmut Mix, Wolfgang E. Nagel, (Felix Schmitt), Robert Schöne, Jonas Stolle, Ronny Tschüter, (Michael Wagner), Matthias Weber, Bert Wesarg, (Thomas William), Frank Winkler, Johannes Ziegenbalg