AutoTune Goals

- Automatic application tuning
  - Performance and energy
- Parallel architectures
  - HPC and parallel servers
  - Homogeneous and heterogeneous
  - Multicore and GPU accelerated systems
  - Reproducable execution capabilities
- Variety of parallel paradigms
  - MPI, HMPP, parallel patterns
Periscope
Performance Analysis Toolkit

- **Online**
  - no need to store trace files
- **Distributed**
  - reduced network utilization
- **Scalable**
  - Up to 100000s of CPUs
- **Multi-scenario analysis**
  - Single-node Performance
  - MPI Communication
  - OpenMP
- **Portable**
  - Fortran, C with MPI & OMP
  - Intel Itanium2, x86 based systems
  - IBM Power6, BlueGene P, Cray

Graphical User Interface (GUI)
Interactive Frontend
Performance Analysis Agent Network
  - Master Agent
  - Communication Agent
  - Analysis Agent
MRI
Application with Monitor

http://www.lrr.in.tum.de/periscope

www.autotune-project.eu
Autotune Approach

- Predefined tuning strategies combining performance analysis and tuning
- Plugins
  - Compiler based optimization
  - HMPP tuning for GPUs
  - Parallel pattern tuning
  - MPI tuning
  - Energy efficiency tuning
Compiler Flag Selection Plugin

• Single node performance is of utmost importance for overall performance and energy efficiency.
• Modern compilers have heuristics for application tuning that might not be optimal.
• The plugin allows to automatically evaluate compiler flags based on expert knowledge.
• Features:
  – Automatic recompilation
  – Multiple plugin search strategies
  – Selective compilation of compute intensive files
  – Flexible specification of compiler flags to explore
The HMPP Tuning Plugin for GPUs aims to optimize HMPP Codelet computations on accelerators by adjusting kernel execution time to the many-core architecture.

**Goal**
- Optimization of HMPP Codelet computations on accelerators
- Adjust kernel execution time to the many-core architecture

**Tuning Points**
- Selection of a static codelet variant
- Modification of runtime conditions
  - GPU grid size, ...
- Tuning points are set at an HMPP callsite location

**Tuning Actions**
- Select codelet variant

**Prepare for tuning**
- The user provides statically a set of codelet variants
- Generate information on the location of callsite regions in the code & related tuning points for codelet variants and runtime configuration

**Tuning results**
- Retrieve the codelet variant and the runtime configuration with the minimal execution time
Tuning Plugin of High-Level Pipeline Patterns for CPU/GPU

- **High-Level Component-based Parallelization**
  - Multi-architectural components
  - Asynchronous, task-based execution model
  - Intelligent runtime system (StarPU)
  - cf. European PEPPHER project

- **Tuning of pipeline structure**
  - OO Coordination Layer
  - Replication factors, buffer sizes

- **Tuning of runtime properties**
  - StarPU scheduling policies
  - Number of CPUs and GPUs

```cpp
#pragma pph pipeline buffer(UNORDERED, ?)
while ( inputstream >> file ) {
    ReadImage(file, image);
    ResizeAndColorConvert(image, outimage);
    #pragma pph stage replication(2:10:2)
    DetectFace(outimage);
    WriteFaceDetectedImage(file, outimage);
}
```

Image processing pipeline with user-provided hints

Integration of the Pipeline Coordination Layer with PTF
MPI Tuning Plugin

- MPI parameters
  - Easy to automate tuning
  - Many parameters available
    - MP_COLLECTIVE_OFFLOAD
    - MP_USE_BULK_XFER
    - MP_EAGER_LIMIT

- Master/Worker applications
  - Easy to automate tuning
  - Number of workers
  - Data partition factor
Energy Efficiency Tuning Plugin

Tuning Plugin for Energy Consumption via CPUFreq using enopt library (libenopt)

• **Aim**
  – Optimize the energy consumption of an arbitrary application, by choosing the best combination of CPUFreq parameters for each code region.

• **Integration with periscope**
  – The start of each code region calls (per callback) the corresponding libenopt function to change:
    • The CPU governor
    • The CPU frequency
  – The code is executed for each combination of frequencies and governors, looking for the minima energy consumption.
Periscope Tuning Framework

• Online
  – Analysis and evaluation of tuned version in single application run
  – Multiple versions in single step due to parallelism in application

• Result
  – Tuning recommendation
  – Adaptation of source code and/or execution environment
  – Impact on production runs
Expected Impact

• Improved performance of applications
• Reduced power consumption of parallel systems
• Facilitated program development and porting
• Reduced time for application tuning
• Leadership of European performance tools groups
• Strengthened European HPC industry
Partners

Technische Universität München

Universität Wien

CAPS Entreprises

Universitat Autònoma de Barcelona

Leibniz Computing Centre

National University of Galaway, ICHEC
Continuous live demonstrations

- At booth #3514 (University of Vienna) and booth #3241 (Leibniz Supercomputer Centre)

- Schedule:

<table>
<thead>
<tr>
<th>Day</th>
<th>Time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuesday November 12th</td>
<td>1:00 pm</td>
<td>Booth #3514 University of Vienna</td>
</tr>
<tr>
<td></td>
<td>5:00 pm</td>
<td>Booth #3241 LRZ</td>
</tr>
<tr>
<td>Wednesday November 13th</td>
<td>1:00 pm</td>
<td>Booth #3514 University of Vienna</td>
</tr>
<tr>
<td></td>
<td>5:00 pm</td>
<td>Booth #3241 LRZ</td>
</tr>
<tr>
<td>Thursday November 14th</td>
<td>1:00 pm</td>
<td>Booth #3514 University of Vienna</td>
</tr>
</tbody>
</table>