Applying Distributed Parametric Optimization Algorithms to Partial Wave Analysis

Dr. Rüdiger Berlich
ISGC 2011 / March 22, 2011
Proposal:

„Automated parametric optimization is defined as the search for the best accessible parameter set that, for a given, computer-implemented evaluation criterion $f$, satisfies a number of predefined constraints“
Some problems have very complex and long-running evaluation functions
Special algorithms are needed that avoid having to visit large portions of the parameter space

Even better, if they can be easily parallelized …
Evolutionary Algorithms / Rastrigin Function

Rastrigin / iteration 0 / fitness = 76.7586

Done with Geneva; Plot created with the ROOT framework
Evolutionary Algorithms / Rastrigin Function

Rastrigin / iteration 1 / fitness = 19.7801

Done with Geneva; Plot created with the ROOT framework
Evolutionary Algorithms / Rastrigin Function

Rastrigin / iteration 2 / fitness = 10.0394

Done with Geneva; Plot created with the ROOT framework
Evolutionary Algorithms / Rastrigin Function

Rastrigin / iteration 3 / fitness = 4.56426

Note: Quick convergence, but bad performance close to global optimum!

Done with Geneva; Plot created with the ROOT framework
Setting the Scene

Geneva

(Grid-enabled evolutionary algorithms)

- Parallel optimization of problems from scientific and industrial domains
- Covers multi-core machines, clusters, Grids and Clouds
- Implemented in portable C++
- Version 0.91 was released recently (cmp. http://www.launchpad.net/geneva)
- Open Source: Covered by the Affero GPL v3
- Subject to a spin-off from KIT (see http://www.gemfony.com)
What it does

- Geneva implements four optimization algorithms
  - Evolutionary Algorithms
  - Swarm Algorithms
  - Gradient Descents
  - Simulated Annealing

- Focus on long-lasting, computationally expensive evaluation functions
  - Stability of core library rated higher than efficiency
  - Parallel execution in multi-threaded mode
  - Suitable for distributed environments, (Clusters, Grids and Clouds)

- Fault tolerance built into the software
  - Cannot rely on continuous availability of machines in networked environments
Optimization problems can be found in just about every field of engineering, natural sciences as well as business and economic sciences (and every other part of life)
PWA Strategy

- Goal: extraction of the complex transition amplitude $\mathcal{M}(k_1, \ldots, k_n)$

\[ \mathcal{M}(k_1, \ldots, k_n) = C_1 A_{\text{res1}}(k_1, \ldots, k_n) + C_2 A_{\text{res2}}(k_1, \ldots, k_n) + \ldots \]

- $\mathcal{M}(k_1, \ldots, k_n)$: (in)coherent sum of the amplitudes of all individual intermediate resonances

- Probability to find an event at a certain point in the phase space
  \[ \sim |\mathcal{M}(k_1, \ldots, k_n)|^2 \rightarrow \text{measurement} \]

PWA: fitting experimental data and MC data to obtain the weights $C_1, C_2, \ldots$

Slide provided by Dr. Bertram Kopf of Ruhr-Universität Bochum
PWA Strategy

- Example: $\bar{p}p \rightarrow \omega \pi^0 \rightarrow (\pi^0 \gamma) \pi^0$

- Determination of the different $J^{PC}$ contributions
- Point in the phasespace defined by 2 properties
  - $\omega$ production angle
  - $\omega$ decay angle
- PWA: find best fit parameters to describe the angular distribution
PWA Strategy: \( \bar{p}p \rightarrow \omega \pi^0 \rightarrow (\pi^0 \gamma) \pi^0 \)

- Fits with different largest contributing spin \( J_{\text{max}} \)

- Amplitudes calculated in the helicity formalism

- Criteria for the largest contributing spin
  - fit result (logLH)
  - obtained weighted angular distributions (\( \omega \) production and decay)
Fit Result for $\bar{p}p \rightarrow \omega\pi^0$ @ 600 MeV/c

Angular Distribution for $J_{\text{max}} = 2$

Angular Distribution for $J_{\text{max}} = 3$

$\omega$ production angle

$\omega$ decay angle

$\cos(\theta)$

entries [a.u.]

data

weighted MC

Entries 5535
Mean 0.054096
RMS 0.6434

Entries 6829
Mean 0.02665
RMS 0.6059

Entries 5535
Mean 0.054096
RMS 0.6434

Entries 6829
Mean 0.02665
RMS 0.6059
I am not an expert for Partial Wave Analysis!
  • But I don't have to be …
  • Optimization algorithms are virtually generic, after all

This use case is ideally suited for distributed computation, due to long evaluation cycles and a single target criterion

Our tests use the Pawian package for the evaluation step
  • Data from the Crystal Barrel experiment
  • Contributors to Pawian include:
    - Miriam Fritsch (Mainz), Albrecht Gillitzer (Jülich), Fritz-Herbert Heinsius (RUB), Klaus Götzen (GSI), Bertram Kopf (RUB), Mathias Michel (Mainz), Sebastian Neubert (Munich), Klaus Peters (GSI), Mathias Steinke (RUB)
  • Special Thanks!
Performance

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>PR</th>
<th>NI</th>
<th>VIRT</th>
<th>RES</th>
<th>SHR</th>
<th>S</th>
<th>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>25879</td>
<td>berlich</td>
<td>17</td>
<td>0</td>
<td>2684m</td>
<td>668m</td>
<td>28m</td>
<td>S</td>
<td>4803.0</td>
<td>0.7</td>
<td>97:24.42</td>
<td>GOmegaPiApp</td>
</tr>
<tr>
<td>25868</td>
<td>berlich</td>
<td>15</td>
<td>0</td>
<td>13140</td>
<td>1492</td>
<td>820</td>
<td>R</td>
<td>5.8</td>
<td>0.0</td>
<td>0:02.31</td>
<td>top</td>
</tr>
<tr>
<td>25603</td>
<td>berlich</td>
<td>15</td>
<td>0</td>
<td>90128</td>
<td>1824</td>
<td>1088</td>
<td>S</td>
<td>0.0</td>
<td>0.0</td>
<td>0:00.08</td>
<td>sshd</td>
</tr>
<tr>
<td>25604</td>
<td>berlich</td>
<td>15</td>
<td>0</td>
<td>66060</td>
<td>1620</td>
<td>1252</td>
<td>S</td>
<td>0.0</td>
<td>0.0</td>
<td>0:00.05</td>
<td>bash</td>
</tr>
<tr>
<td>25653</td>
<td>berlich</td>
<td>15</td>
<td>0</td>
<td>90128</td>
<td>1828</td>
<td>1098</td>
<td>S</td>
<td>0.0</td>
<td>0.0</td>
<td>0:00.06</td>
<td>sshd</td>
</tr>
<tr>
<td>25654</td>
<td>berlich</td>
<td>15</td>
<td>0</td>
<td>66060</td>
<td>1596</td>
<td>1240</td>
<td>S</td>
<td>0.0</td>
<td>0.0</td>
<td>0:00.01</td>
<td>bash</td>
</tr>
</tbody>
</table>

- 48 Core AMD / Magni Cours
- Uses Evolutionary Strategies, as implemented in Geneva
- Program has Minuit Mode, consumes in average only approximately half of the available compute time

Special thanks to the department „Scientific Computing Labs“ of Steinbuch Centre for Computing / KIT, for letting me use this machine!
Scalability in a network

- Curve uses a toy example, whose evaluation function could be arbitrarily scaled
- Performing PWA with ES on 100 cores in parallel in a network takes approximately 2 hours for JMax = 5, 1800 MeV/c
- Minuit in MT mode on 48 core system didn't finish after 7 hours
Scalability: The 80/20 rule (low hanging fruit!)

Fit results with Geneva

\[ \bar{p}p \rightarrow \omega \pi^0 \]

1800 MeV/c
JMax = 5
Crystal Barrel Data
Fit results with Geneva in ES mode

\[ p\bar{p} \rightarrow \omega \pi^0 \]

1800 MeV/c

Crystal Barrel Data

Better value possible (approx. -450)
Analysis

- From the optimization engineer's perspective: The procedure seems to be fragile
  - Starts with approx 100,000, searches for differences between optima of a few 100
  - It seems necessary to perform each fit several times with varying starting points to confirm final result
  - Will highly benefit from parallelization, due to high demand for computing power
- Tests of Pawian group (particularly Mathias Michel + Denis Pevlina) confirmed that best results are achieved when combining different algorithms
  - E.g.: „raw optimization“ with ES, „last mile“ with GD
  - Geneva now provides Swarm, Gradient Descent and Simulated Annealing alongside Evolution Strategies
  - But still needs work. Best option at the moment appears to be combination of Geneva ES + Minuit
  - Will soon be able provide „one-stop“ solution, as individuals can move freely between optimization algorithms in Geneva
Question ? Questions!

ruediger.berlich@kit.edu
http://ruediger.berlich.com