From Libraries to Workflows

Bioinformatics with KNIME

Generic Knime Node Generator and CTD

OpenMS – Proteomics/Metabolomics
SeqAn – Next generation sequencing
BALL – Structural Bioinformatics

Oliver Kohlbacher, Eberhard-Karls Universität Tübingen
Knut Reinert, Freie Universität Berlin
SeqAn is a **generic** open source C++ library of **efficient** algorithms and data structures for the analysis of NGS data

[www.seqan.de](http://www.seqan.de)

OpenMS is an open-source software C++ library for **proteomics** and **metabolomics** data analysis (together with Tübingen and ETH Zurich)

[www.openms.de](http://www.openms.de)

**Generic KNIME Nodes** – generating KNIME nodes from CTD files. CTD - an XML-based format to describe command line tools.
Integrated in OpenMS, SeqAn and BALL.
Software libraries as intermediate

Experimentalists

Algorithm libraries

Computer Scientists

Analysis pipelines

Maintainable tool

Prototype implementation

Algorithm design

Theoretical Considerations
Workflows Enabling Software Re-Use

Data repositories (e.g. OpenBIS)

Workflow Engine to design and execute

External Computing (cloud, grid, cluster)

Distribution Platform (e.g. KNIME site)

Node generator using CTDs

Transcriptomics
Genomics
Metagenomics

Proteomics Metabolomics

Computer-aided drug design

External tools
BWA, tophat, EMBOS, SAMTOOLS, etc.
A short introduction to OpenMS
Data Flow in Shotgun Proteomics

Sample → HPLC/MS → Raw Data

Maps → Data Reduction → Peak Data

Annotated Maps → Identification → Differentially Expressed Proteins

50 MB → 1 GB → 50 kB
OpenMS

Software framework for shotgun proteomics

ISO/ANSI C++ compliant

Features

- Open Source (LGPL [1.9]/BSD [1.10])
- Data structures & algorithms
  - Core data structures for MS
  - Quantification and identification
  - A total of 172 tools
- Data import/export in standard formats
- Database storage of all data
- Visualization for spectra, maps
- Currently: ~ 550,000 lines of code
TOPP tools

Documentation for each tool is available as part of the OpenMS documentation (www.OpenMS.de)

FeatureFinder

The feature detection application for quantitation.

Potential predecessor tools: PeakPicker, MapAligner
Potential successor tools: FeatureLinker, SeedListGenerator

This module identifies "features" in a data map by features. The algorithm detects peaks in a LC-MS sample that reveals a characteristic isotope distribution. The algorithm computes positions in rt and m/z dimension and a charge estimate of each peptide.

The algorithm identifies pronounced regions of the data around so-called seeds. In the next step, we iteratively fit a model of the isotope profile and the retention time to these data points. Data points with a low probability under this model are removed from the feature region. The intensity of the feature is then given by the sum of the data points included in its regions.

How to find suitable parameters and details of the different algorithms implemented are described in the TOPP tutorial.

Note:
that the wavelet transform is very slow on high-resolution spectra (i.e. FT, Orbitrap). We recommend to use a noise or intensity filter to remove spurious points first and to speed-up the feature detection process.

Specialized tools are available for some experimental techniques: SILACAnalyzer, ITRAQAnalyzer.

The command line parameters of this tool are:

```
FeatureFinder -- Detects two-dimensional features in LC-MS data.
Version: 1.7.0 Sep 3 2010, 15:13:04, Revision: 7349
Usage:
  FeatureFinder <options>
```
Documentation

Documentation for each tool is available as part of the OpenMS documentation (www.openms.de)

Common TOFP options:

  -ini <file>      Use the given TOFP INI file
  -threads <n>     Sets the number of threads allowed to be used by the TOFP tool (default: "1")
  -write_ini <file> Writes the default configuration file
  --help           Shows options
  --helphelp       Shows all options (including advanced)

The following configuration subsections are valid:
  - algorithm     Algorithm section

You can write an example INI file using the ‘-write_ini’ option. Documentation of subsection parameters can be found in the doxygen documentation or the INIFileEditor. Have a look at OpenMS/doc/index.html for more information.

For the parameters of the algorithm section see the algorithms documentation:

  centroided
  isotope_wavelet
  mrm

In the following table you can find example values of the most important parameters for different instrument types. These parameters are not valid for all instruments of that type, but can be used as a starting point for finding suitable parameters.

'centroided' algorithm:

<table>
<thead>
<tr>
<th></th>
<th>Q-TOF</th>
<th>LTQ Orbitrap</th>
</tr>
</thead>
<tbody>
<tr>
<td>intensity:bins</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>mass_trace:mz_tolerance</td>
<td>0.02</td>
<td>0.004</td>
</tr>
<tr>
<td>isotopic_pattern:mz_tolerance</td>
<td>0.04</td>
<td>0.005</td>
</tr>
</tbody>
</table>

For the centroided algorithm centroided data is needed. In order to create centroided data from profile data use the PeakPicker.
PSI Standard Formats

Numerous open and standardized XML formats have been proposed by the HUPO Proteomics Standards Initiative (HUPO PSI):

- **mzML** for storing mass spectrometry data
- **mzIdentML** for storing peptide/protein identifications
- **TraML** for storing transition lists and inclusion lists
- **mzQuantML** for storing quantitation results (in revision)
- **mzTab** for summary information of quantitative and qualitative results, Excel-friendly (in revision)
- **qcML** for quality control data (in preparation)
- OpenMS internal formats: idXML, featureXML, consensusXML – to be replaced by PSI formats

**Advantages**

- Open, documented
- No closed-source libraries required
- Will still be readable in 10 years from now
- Interoperable with different software packages

**Disadvantages**

- Conversion required (and often awkward)
- File size
- Poor support by instrument vendors
A short intro to SeqAn
Seqan in a nutshell

SeqAn is an open source C++ library of efficient algorithms and data structures for the analysis of sequences with the focus on biological data.

SeqAn applies a unique generic design that guarantees high performance, generality, extensibility, and integration with other libraries.

SeqAn is easy to use, and it simplifies the development of new software tools with a minimal loss of performance.
SeqAn developers

External
CSC
BMBF
DFG
IMPRS
FU
Library Contents

Sequences
- strings
- structured sequences
- gapped sequences
- alterators

Searching
- exact/approximate
- search heuristics
- filtering
- motif search

Probabilis.
- profiles, weight matrices
- HMM, SCFG
- p-value computations
- ...

Alignments
- alignment data structures
- dynamic Programming
- alignment heuristics
- multidimensional chaining

Indices
- q-gram hashes
- (enhanced) suffix array
- suffix trees
- lazy indices, compress. ind.

Graphs
- (structural) align. graphs
- word graphs
- probabilistic automata
- trees

Algorithms
- RazerS, Masai
- gQUASAR, SWIFT,…
- MEME, PROJECTION,…
- ...

Integration
- using external tools
- STL
- Lemon Graph library
- friend libraries

Biologicals
- alphabets
- scoring schemes
- file formats
- base pair probabilities

Miscellan.
- allocators
- OS access and support
- helper data struc. and algorithms
Collaborations

We collaborate with different academic and industrial Groups, among them:

- Collaboration (Jacopo Pantaleoni) to parallelize indices and verification algorithms in SeqAn, to speed up any applications making use of indices.

- Collaboration (Dirk Evers). NYGC will develop tools based on SeqAn.

- Collaboration (Tony Cox) to replace parts of the Grouper pipeline.
Generic KNIME nodes
Library Integration

• Give every tool a self-describing output format: semantic annotation of its inputs/outputs
• In OpenMS and SeqAn we developed CTD (Common Tool Description) for this purpose

• Each tool can thus ‘tell’ its requirements and options in a coherent format
• All interfaces are fully described by this format
• Information on the tools options, I/O are entirely contained within the individual tool (maintenance!)
General tool description in header

<tool>
<name>NoiseFilterSGolay</name>
<version>1.9.0</version>
<description>Removes noise using a Savitzky Golay filter.</description>
<manual>[…]</manual>
<docurl>http://openms.de/1.9/doc/NoiseFilterSGolay.html</docurl>
<category>Signal processing and preprocessing</category>
...
...
</tool>
CTD Format

Parameter (including type, description, valid ranges) and I/O format description (mappable to MIME types)

<tool>
  ...
  <ITEM name="in" value="" type="string" description="input raw data file"
    tags="input file,required" restrictions="*.mzML" />
  <ITEM name="out" value="" type="string" description="output raw data file"
    tags="output file,required" restrictions="*.mzML" />
  <NODE name="algorithm" description="Algorithm parameters section">
    <ITEM name="frame_length" value="11" type="int"
      description="number data points used for smoothing." />
    <ITEM name="polynomial_order" value="4" type="int"
      description="Order of the polynomial that is fitted." />
  </NODE>
  ...
</tool>
External Tools

External tools cannot produce a CTD description of themselves. However, the CTD format allows to include information to generate the appropriate command line.

```xml
<cli>
  <clielement optionIdentifier="-sequence">
    <mapping referenceName="banana.sequence"/>
  </clielement>
  <clielement optionIdentifier="-outfile">
    <mapping referenceName="banana.outfile"/>
  </clielement>
</cli>

<PARAMETERS version="1.3">
  <NODE description="Bending and curvature plot in B-DNA" name="banana">
    <ITEM name="sequence" restrictions="data" tags="input file,required" type="string"/>
    <ITEM name="outfile" restrictions="txt" tags="output file,required" type="string"/>
  </NODE>
</PARAMETERS>
```
Node generator

Generic KNIME Nodes can generate nodes to be used in KNIME.

https://github.com/genericworkflownodes

It is compatible with both internal and external tools. This means, ANY tool can be integrated in KNIME as long as it has a CTD.
Generic KNIME Nodes

The following folder structure is needed:

```
<base folder>
  descriptors/
    tool1_ctd.xml
    tool2_ctd.xml
  icons/
    category.png
    splash.png
    plugin.properties
```
Integrating CTD Tools in KNIME
GKN generates Nodes

<base folder>
  descriptors/
    tool1_crd.xml
    tool2_crd.xml
  icons/
    category.png
    splash.png
    plugin.properties

GKN

JAR

com.foo.tools.jar

KNIME
Other GKN Features

It also provides with Input and Output File nodes, File to Table, Table to File conversion nodes, and other utilities.
GKN workflow (OpenMS)

small peak picking workflow
Wrapping External Tools
Integrating CTD Tools in KNIME

No more scripting
Workflows Enabling Software Re-Use

External tools

 automatice

manuel

program

script

Galaxy

STL

OPENMS

SEQAN

GKN

KNIME

gUSE
Exporting KNIME Workflows

• Sometimes you just need more computing power.

• It would be nice to be able to deploy a KNIME workflow prototyped on your laptop on a grid
Exporting to gUSE / WSPGRADE
Exporting to gUSE / WSPGRADE

Select the destination format

Select an export format: **gUse / WS-PGRADE**

Select an export destination: **export.zip**

Converting KNIME workflows in a format understood by gUse/WS-PGRADE managed grids.
Exporting to gUSE / WSPGRADE
Exporting to gUSE / WSPGRADE

Concrete

Submit All  Refresh

<table>
<thead>
<tr>
<th>Workflow name</th>
<th>Workflow type</th>
<th>Submitted</th>
<th>Running</th>
<th>Finished</th>
<th>Error</th>
<th>Suspended</th>
</tr>
</thead>
<tbody>
<tr>
<td>KNIME_Export1362346410783 zen</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Message: KNIME can do it!
Exporting to gUSE / WSPGRADE

Workflow name: KNIME_Export1362346410753
Note: KNIME can do it!
Workflow Graph: KNIME_Export1362346410753
Workflow Template:
Case studies
OpenMS Case study: Consensus Identification

Identification with different search engines

<table>
<thead>
<tr>
<th>Rank</th>
<th>Peptide</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>QRESTATDILQK</td>
<td>0.008</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>Peptide</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EIEEDSLEGKK</td>
<td>14.78</td>
</tr>
<tr>
<td>2</td>
<td>GIEDDLMDLKK</td>
<td>12.63</td>
</tr>
<tr>
<td>3</td>
<td>ISCAEGALEALKK</td>
<td>10.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>Peptide</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AELASCVVGDLGAK</td>
<td>1.2</td>
</tr>
<tr>
<td>2</td>
<td>ELM(Ox)SNGPGSII GAK</td>
<td>1.2</td>
</tr>
<tr>
<td>3</td>
<td>ISCAEGALEALKK</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>QRESTATDILQK</td>
<td>10</td>
</tr>
</tbody>
</table>

Consensus Scoring

Score for every sequence from any engine

<table>
<thead>
<tr>
<th>Rank</th>
<th>Peptide</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>QRESTATDILQK</td>
<td>0.54</td>
</tr>
<tr>
<td>2</td>
<td>EIEEDSLEGLKK</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>GIEDDLMDLIKK</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ISCAEGALEALKK</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>AELASCVVGDGLGAK</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>ELM(Ox)SNGPGSII GAK</td>
<td></td>
</tr>
</tbody>
</table>

Combination of scores

\[
\text{ConsensusID (p1)} = \frac{s_1(p_1) + \alpha s_2(p_i) + \beta s_3(p_j)}{(1 + \alpha + \beta)^2}
\]

\[
\text{ConsensusID (QRESTATDILQK)} = \frac{0.54 + 0.3 \cdot 0.96 + 1 \cdot 0.99}{(1 + 0.3 + 1)^2} = 0.34
\]

In the following I present a first application based on a collaboration with the Robert Koch Institute in Berlin based on

Quantitative analysis

Sequencing gives N reads (N can be $10^9$)
Problem

What species are present?
With what abundance?

Answer:

• **Simple:** If the genomes are quite different
• **Hard:** if the genomes are quite similar
Read mapping (e.g. bwa)

- Red bars: True abundance $c_1 = r_1 = 0.24$
- Blue bars: True abundance $c_2 = r_2 = 0.22$
- Green bars: True abundance $c_3 = r_3 = 0.52$

$c$: true abundance
$r$: observed abundance
Hard

Read mapping (e.g. bwa) observed abundance $r$ unequal with true abundance $c$

Abundance

$r_1 = 0.49$
$r_2 = 0.47$
$r_3 = 0.52$
Similarity matrix

\[
\begin{array}{ccc}
S_1 & S_2 & S_3 \\
S_1 & 1 & 0.95 & 0.01 \\
0.24 & 0.25 & + & 0 \\
S_2 & 0.93 & 1 & 0.02 \\
S_3 & 0.01 & 0.02 & 1 \\
\end{array}
\]

Abundance

\[
\begin{align*}
    r_1 &= 0.49 \\
    r_2 &= 0.47 \\
    r_3 &= 0.52 \\
\end{align*}
\]

\[
\sum_j a_{ji}c_j = r_i.
\]

\[
Ac = r.
\]
Solution

\[ C = A^{-1} r. \]

- Matrix inversion is numerically unstable
- Negative abundances possible
- Constraint error minimization with values between 0 und 1.

\[ \hat{c} = \arg \min_C ||Ac - r||_2 \]

s.t. \( \hat{c}_i \geq 0 \ \forall i \) and \( \sum_i |\hat{c}_i| \leq 1 \)

Lindner, M.S. and Renard, B. Y. (2012)
## Similarity matrix

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1$</td>
<td>1</td>
<td>0.95</td>
<td>0.01</td>
</tr>
<tr>
<td>$S_2$</td>
<td>0.93</td>
<td>1</td>
<td>0.02</td>
</tr>
<tr>
<td>$S_3$</td>
<td>0.01</td>
<td>0.02</td>
<td>1</td>
</tr>
</tbody>
</table>

$A=(a_{ij})$ is estimated as follows:

- Sample reads u.i.d from all genomes
- Map the reads with the **same** program (e.g. bwa) as used for the real input
- Use the portion of reads from genome $i$ that map to genome $j$ as estimator for $a_{ij}$
Varroa destructor virus
CTD integration in releases

- **OpenMS 1.10** (released **today**) has a full integration of CTD generation and KNIME support
- **SeqAn 1.4** (release this spring)
6th OpenMS User Meeting
September 3-5, 2013, ETH Zürich

- Proteomics and metabolomics case studies with OpenMS
- New developments in OpenMS
- Advanced and beginner tutorials
- Doctor’s corner – bring your own data
- Keynote by Ruedi Aebersold
- OpenMS.de/UM2013
SeqAn and the VIP Biostore

3rd Project Workshop, September 2013

FU Berlin, Institute for Computer Science

Details will be soon on the website www.seqan-biostore.de
THANK YOU

Luis de la Garza, Tübingen

Stephan Aiche, Berlin

Marc Röttig
The OpenMS and SeqAn teams
The KNIME team