Galaxy to WS-PGRADE

- Galaxy workflows in Python dictionary and JSON (JavaScript Object Notation)
- WS-PGRADE workflows in XML
- Visualization completely different
  - Map Python dictionary to XML tags
  - Calculate coordinates via a graph algorithm
Galaxy to WS-PGRADE

- Galaxy workflows in Python dictionary and JSON
- WS-PGRADE workflows in XML
- Visualization completely different

⇒ Map Python dictionary to XML tags
⇒ Calculate coordinates via a graph algorithm
Web-based Science Gateways for Structural Bioinformatics

- Galaxy workflows in Python dictionary and JSON (JavaScript Object Notation)
- WS-PGRADE workflows in XML
- Visualisation completely different

⇒ Map Python dictionary to XML tags
⇒ Calculate coordinates via a graph algorithm

Graph of the workflow to be configured
Graph Algorithm

- Workflows can be represented by directed graphs
- Graph $G = (V, E)$ consisting of a finite set of vertices $V$ and a set of edges $E$ with $E \subseteq V \times V$
- Topological sort
- Tarjan’s algorithm
- Linear running time $O(|V| + |E|)$

1. A datastructure for a directed graph is created
2. All root elements are filtered as starting point for Task 3
3. All root elements are added to a list as long as a root element exists

Graph Algorithm

V = a, b, c, d, e, f
E = (a,b), (c,d), (a,e), (b,f), (d, e), (e,f)

Topological sort
1. Sorted list (a, c), root = b, d
2. Sorted list (a, c, b, d), root = e
3. Sorted list (a, c, b, d, e), root = f
4. Sorted list (a, c, b, d, e, f)
Graph Algorithm

$V = a, b, c, d, e, f$

$E = (a,b), (c,d), (a,e), (b,f), (d, e), (e,f)$

Topological sort

1. Sorted list $(a, c)$, root = $b, d$
2. Sorted list $(a, c, b, d)$, root = $e$
3. Sorted list $(a, c, b, d, e)$, root = $f$
4. Sorted list $(a, c, b, d, e, f)$

Graph Algorithm

V = a, b, c, d, e, f
E = (a,b), (c,d), (a,e), (b,f), (d, e), (e,f)

Topological sort
1. Sorted list (a, c), root = b, d
2. Sorted list (a, c, b, d), root = e
3. Sorted list (a, c, b, d, e), root = f
4. Sorted list (a, c, b, d, e, f)

Graph Algorithm

V = a, b, c, d, e, f
E = (a,b), (c,d), (a,e), (b,f), (d, e), (e,f)

Topological sort
1. Sorted list (a, c), root = b, d
2. Sorted list (a, c, b, d), root = e
3. Sorted list (a, c, b, d, e), root = f
4. Sorted list (a, c, b, d, e, f)

Graph Algorithm

$E = (a,b), (c,d), (a,e), (b,f), (d, e), (e,f)$

Sorted list \((a, c, b, d, e, f)\)
Graph Algorithm

E = (a,b), (c,d), (a,e), (b,f), (d, e), (e,f)
Sorted list (a, c, b, d, e, f)

Calculation of coordinates
Graph Algorithm

\[ E = (a,b), (c,d), (a,e), (b,f), (d,e), (e,f) \]

Sorted list \((a, c, b, d, e, f)\)

**Calculation of coordinates**
Graph Algorithm

$E = (a,b), (c,d), (a,e), (b,f), (d,e), (e,f)$

Sorted list $(a, c, b, d, e, f)$

Calculation of coordinates
Graph Algorithm

E = (a,b), (c,d), (a,e), (b,f), (d, e), (e,f)
Sorted list (a, c, b, d, e, f)

Calculation of coordinates
Graph Algorithm

$E = (a,b), (c,d), (a,e), (b,f), (d,e), (e,f)$
Sorted list $(a, c, b, d, e, f)$

Calculation of coordinates
Graph Algorithm

E = (a, b), (c, d),...

Sorted list (a, c, b, d, e, f)

Calculating coordinates
Graph Algorithm

Avoid intersections at same vertex

\[
\text{area}(a, b, c) := \begin{vmatrix}
  a_x & a_y & 1 \\
  b_x & b_y & 1 \\
  c_x & c_y & 1
\end{vmatrix} = (c_y - a_y)(b_x - a_x) - (b_y - a_y)(c_x - a_x)
\]

\[
\text{area}(a, b, c) \begin{cases} 
  < 0 & \text{if } a, b, c \text{ in clockwise angle} \\
  = 0 & \text{if } a, b, c \text{ collinear} \\
  > 0 & \text{if } a, b, c \text{ in counter clockwise angle}
\end{cases}
\]

a, b, c, d intersect if
ccw(a, c, d) ≠ ccw(b, c, d) and
ccw(a, b, c) ≠ ccw(a, b, d)
Graph Algorithm

Avoid intersections at the same vertex

\[
\text{area}(a, b, c) := \begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix} \\
\begin{cases} < 0 & \text{if } a, b, c \text{ are counterclockwise} \\ = 0 & \text{if } a, b, c \text{ are collinear} \\ > 0 & \text{if } a, b, c \text{ are clockwise} \end{cases}
\]
Sustainability

MoSGrid (ends 31.12.2012) but partners participate in

SCI-BUS (SCIlentific gateway Based User Support)
• EU project 01.10.2011 – 30.09.2014
• Extension of the MoSGrid portal with an interactive molecule editor based on WebGL and a semantic search

ER-flow (Building an European Research Community through Interoperable Workflows and Data)
• EU project 01.10.2012 – 30.09.2014
• Integration of applications in SHIWA simulation platform
• Study of data exchange between workflow systems
• Community management
Acknowledgements

- Oliver Kohlbacher
- Jens Krüger
- Marc Röttig
- Georg Birkenheuer
- André Brinkmann
- Johannes Schuster
- Richard Grunzke
- René Jäkel
- Ralph Müller-Pfefferkorn
- Tobias Schlemmer
- Sebastian Breuers
- Dirk Blunk
- Lars Packschies
- Klaus-Dieter Warzecha
- Martin Wewior
- Sonja Herres-Pawlis
- Ines Dos Santos Vieira
- Patrick Schäfer
- Thomas Steinke
- Bernd Schuller
- Valentina Huber
- Frank Cordes
- Peter Kacsuk
- Ákos Balaskó
- Miklos Kozlovszky
- István Márton
- Anna Szikszay Fabri
Special Issue on: Distributed, Parallel, and GPU-accelerated Approaches to Computational Biology (COMPBIO2013)

Manuscript Due: February 11, 2013
First Decision Date: March 25, 2013
Revision Due: May 6, 2013
Final Decision Date: June 10, 2013
Final Paper Due: July 15, 2013
Publication Date: 3rd Quarter, 2013 (Tentative)

http://www.cc-pe.net/journalinfo/issues/2013.html
Thank you!
sandra.gesing@uni-tuebingen.de