Graph-Based Linking and Visualization for Legislation Documents (GLVD)

Dincer Gultemen & Tom van Engers

Demand of Parliaments

- Semi-structured information and semantic technologies
  - Inter-institutional business process reduction through interoperability
    - Parliament as harmonizer-hub for the orchestration of inter-institutional law-making process
  - Legal analysis and consistency verification for complexly interlinked regulatory interactions
    - ? More granular and composite network construction
    - ? Multi-layer semantic network verification-analysis

Sources of Law and SNL Model

- More granular and composite network construction
  - Semantically enriched documents
  - The interactions between documents representing sources-of-law = source-network-of-law (SNL)
    - Legislation (statutory law) level network
  - Conceptual graph formation with increased node granularity at the paragraph-article level
  - Verification of SNL = Network Approach + Semantic Technology
- Multi-layer network verification and analysis (Consistency of sources-of-law)
  - The possibility of case-law peripheral to the graph

GLVD

- Data processing pipeline executing SNL model
  - Extendable backbone application
- To detect-annotate interactions and visualize the graph
  - “Text to XML Parsing”
    - Paragraph-article level document linking through self-labeling
  - “XML to Visualization”
    - Arbitrary cyclic-tree graph structure through a semantic XML linking

Network and Law

- Macro-level analysis: the overall structure of legal citation network
  - Fowler et al. (2006) to measure US Supreme Court Opinions (USSCO) case centrality
  - Smith (2007) to measure how skewed the network structure of U.S case law is
  - Boulet et al. (2011) to measure small world indices local-global connectivity for French legal codes and visualization of hemicycle-like plot
  - Winkels et al. (2011) to validate the network approach for the Dutch case law

- Micro-level analysis: logical-semantic aspects and precision in node-link generation
  - Bench- Capon (1997): Consistent and Effective treatment of legal documents is based on the ability to relate documents of different types
    - Document graphs: nodes (meaningful units of textual objects) linked with edges of logical relationships
  - Hamdaqa and Hamou (2011) aiming vertical regulatory compliance (which was also horizontally limited by statutory-administrative level North-American laws and regulations)
    - Graph theory for regulatory dependency patterns (defined-by-defined-by, amend-amend, amend-use, generalized amend-amend.
    - CompDSS (Compliance Decision Support System) was to parse the provisions in hierarchical tree form and the graph was specified as G = (V, E, R) where edges for citations (E) relation type-labeled with (R) between provisions (V)
    - Visualization by the related tool module in DOT scripting language and Graphviz with “neato” approach.

Micro-level Network Construction + Semantic Technologies

- **A Technology for Interoperability**
- Semantic analysis -> irregular, deeply hierarchical and recursive data -> tree-structured information
- The rise of XML -> Resource Description Framework (RDF) to Web Ontology Language (OWL)
- Still XML -> Higher level universality + standardization of XML
- **An Application for Semantics**
- Semantic web=intelligent data rather than intelligent application.
- (XML<RDF<OWL) trade-off
  - RDF : Additional labeling on XML / XML: additional applications (annotator) to play with XML better
- GLVD: Application-level semantic processing with legislative XML + interoperability
  - Once GLVD defines the linking -> RDF/XML serialization is possible
  - Domain specific documents can be processed -> agreed on annotation

Text to XML Parser

- Application Programming Interfaces (APIs) / space (memory)-time (processing) tradeoffs
  - Simple API for XML (SAX) - event driven
  - Document Object Model (DOM) - tree structure (Memory=3xSAX)
  - XML Object Model (XOM): a fast-enough and small-enough parser. (Memory=1.5xSAX)
- Non-rigorous use of self-descriptive elements (delaying CEN Metalex Standard schema)
- Self-Labeling
  - GLVD + (!Reference retrieval patterns repository)
- Assumptions for markup
  - The elements: “Regulation”, “Law”, “Section”, “Part”, “Article”, “Title”, “Paragraph” and “Regulation Date”.
  - Nested directly by upper wrappings is possible: Titles of articles, and sections or part descriptions
  - Granularity at paragraph-article level
  - Self-labeling: Attribute value in natural language defining their structural position within XML tree structure.
    - "Article X of Law number Y” as attribute value
  - Instantiation of there tags,
    - “Section” and “Part” tags are based on empty line (2-1) processing
    - “Article” and “Regulation Date” tags inline processing

Markup requirements

```
<?xml version="1.0" encoding="utf-8"?>
<Regulation ID="r">
  <Law ID="Legislative Document y">
    <Section ID="Section s of Law Number y">
      <Part ID="Part p of Section s of Law Number y">
        <h2 ID="Title Number t1-t2 of Law Number y">TEXT</h2>
        <p ID="Paragraph Number 0 of Article Number x Law Number y">TEXT</p>
        <Article ID="Article x of Law Number y">
          <p ID="Paragraph Number z of Article Number x of Law Number y">TEXT
            <Regulation Date>TEXT</Regulation Date>
          </p>
        </Article>
      </Part>
    </Section>
  </Law>
</Regulation>
```

- One space line after parts - Two space lines after sections
- The lines (500) should contain the entire “Paragraph”
- One character space after “Law Number:”
- No “.” or “;” in the title lines.
- The titles of articles are above the articles
- Descriptions of parts and sections are below the section or part titles.

XML Graph Composition

- GLVD conceptualize SNL in conceptual graph (BG) formation => finite, undirected, and arbitrary cyclic multi-graph
- C (set of concept types) ordered according their types and subtypes being marked by M (set of individual markers), and R (set of relation types) as facts: Assertions to relate entities (concept nodes)
  - type(c), marker(c)=[c:m] || cϵC, mϵM
- G=(C, R, E, I) with labeling function (I) for the edges (E)
  - C=mapping M=>Regulation / Law / Section / Part / Article / h2 / p / Regulation Date
  - R=paragraphHasAttributeValueOf and elementIsTheSubOf relations.
  - Attributes=Legislation Document y / Section x1 / Part x2 / Title Number x3 of Law Number y / Paragraph Number x4 of Law Number y and Article x5 of Law Number y.
- Relation signatures (type(r)) to map a relation symbol (r) to the “maximal concept type” (the most specific inclusive type) of its arguments (σ(r) ϵ Cj for 1≤j≤k) based on the arity (k number of nodes involved) of the relation
  - r=elementIsThe**SubOf**(Element, Element) with mapping
  - σ(r): ([Element:*], [Element:*])
  - r=paragraphHas**ReferenceOf**(Paragraph, Article) with mapping
  - σ(r): ([Element:Paragraph], [Element:Article])
  - Implicit siblingsOf relation (through local variables x and global variable y) => the same level hierarchical mapping

GLVD Visualink Module

- GLVD linking and visualization module is based on two components (Linker and Visualization Component)
- Linker:
  - A parent “file node” is created for XML files,
  - Append each root element to the parent “file node”
    - As there is a file in the folder
  - Rule definitions to create recursive nodes

```
Create a global root element
For all files in the folder
        Parse each file and get the local root element
        Append local root to the global element with a given child name (Attribute Value)
Create a root node for global root element
Create a model: Iterate node generation in a nested way as long as elements has children
If the element is an element having children,
        Read XML child elements
        Create nodes for each child
        Add each created node to the global control list
```

Linker

- Processing and the comparison of the content of created nodes
  - global control list

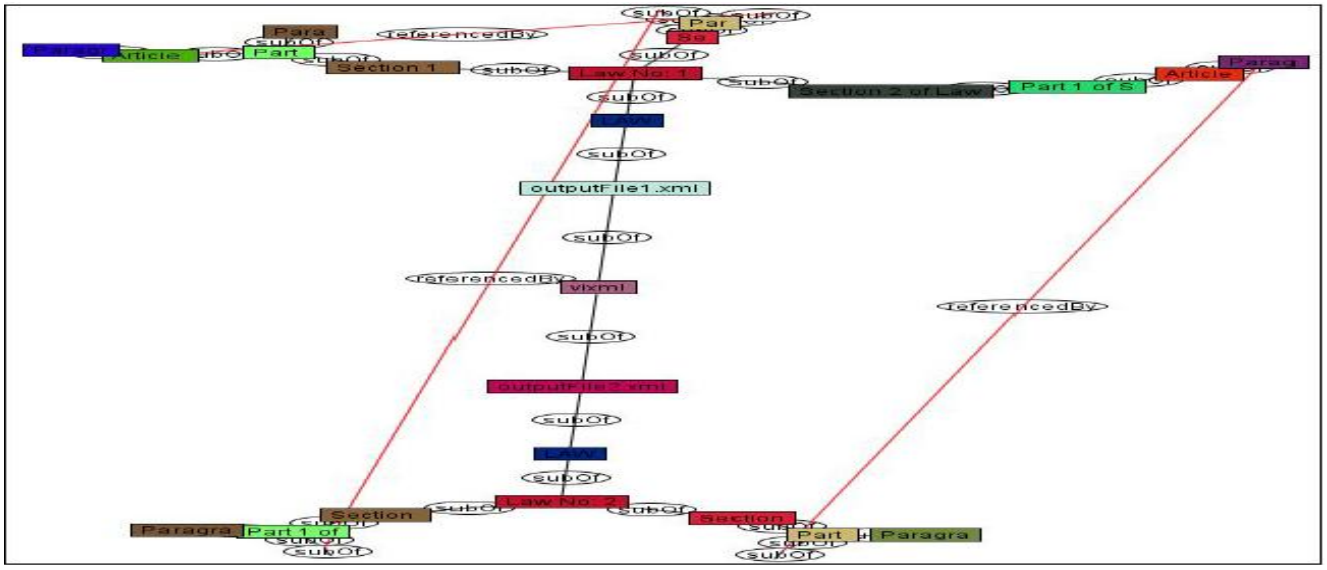
```
Create an empty relation list for explicit reference
Process and compare the content of each possible pair of nodes in global control list
If value of a node contains the name of another node (Attribute Value)
    If this node is a "Paragraph node"
        Add both nodes to the explicit reference list
```

- No nodes except "explicit reference relation"
  - Store and check all children names data (Attribute Values) if children name data contains the node names in explicit reference list.

```
For all nodes in the global control list
    Find children names of each node nested and store this data in the node content
    If children name data does not contain the nodes in the explicit reference list
    Remove the children of that node
```

GLVD Visualization

- XML graph structure and labeling constructed by the linker -> visualization component.
  - Java Foundation Classes (JFC) including APIs like AWT / SWING / Java2D
    - Translation-transformation
    - Translation-animation “move to center”



Future Work

- Experimentation on large scale real SNL graph
- Increased relation types - more semantic features
- Reference retrieval patterns repository
- Algorithm repository (Different jurisdictions)
- CEN Metalex compliance
- Case-law to be included to peripheral to the graph
- Multilevel network analysis-graph based knowledge representation
- SNL and network analysis to estimate the need for legislative action
- More dynamic, user-friendly interface with user-controls

THANKS!

- Questions and Comments???