Practical Language-Independent Detection of Near-Miss Clones

James R. Cordy, Queen's
Thomas R. Dean, Queen's
Nikita Synytskyy, U of Waterloo

Motivation

- Clones occur in any non-trivial application
- Studying clones can help us to understand how programmers use and re-use code
  - **Clones are more than just textually identical pieces of code**
- Near miss clones help identify how code propagates through an application and how it evolves

Clones

- Understanding of clones is useful
  - **Not all clones are bad nor should they all be removed**
- Exact clones have a clear definition
- Near miss clones are more difficult
  - **How close is close?**
  - **Are some changes more important than others?**

Clone Detection (Independent)

- Lexical Approaches [Baker93][Kamiya02]
  - **Substring comparisons**
  - **Independent of structural elements of the language**
- Neural Nets [Davey95]
  - **Language independent**
  - **Requires training sets and training time**

Clone Detection (Dependent)

- Concept Analysis[Kontogiannis96]
  - **Procedure and ADT level**
- AST Based [Baxter98]
- Metrics [Mayrand96]
  - **Procedure level**
- Slicing [Komondoor01]
  - **Finds non-contiguous, but does not handle near miss**
- Dependence Graph [Krinke01][Chen03]
  - **Both syntactic structure and data flow**

Why HTML?

- HTML is a prime candidate for clones
- Tools to reuse HTML are limited
  - **Requires server or client side scripts**
  - **Not really part of HTML itself**
- WYSIWYG tools such as Netscape editor, Dreamweaver and others promote cloning
  - **Copy/paste**

Why HTML?

- Web pages are expected to have a common look and feel (need for common HTML elements)
  - **CSS not sufficient**
- Many web designers have not been exposed to Software Engineering practices
  - **Graphic designers**

2004 Executive

Section Chair	Hisham El-Masry Canadian Electronics Corporation Kinston, ON elmasry@cmc.ca
Past Chair	Thomas R. Dean Department of Electrical and Computer Engineering Queen's University Kingston, ON thomas.dean@ece.queensu.ca
Secretary	Saeed Gazor Department of Electrical and Computer Engineering Queen's University Kingston, ON saeed.gazor@ece.queensu.ca
Treasurer	Subramania Sudharsanan Department of Electrical and Computer Engineering Queen's University Kingston, ON sudha@ece.queensu.ca
Vice-Chair (Technical)	Carlos Saavedra Department of Electrical and Computer Engineering Queen's University Kingston, ON carlos.saavedra@ece.queensu.ca
Vice-Chair (Membership) & Queen's Student Liason	Keyvan Hashtrudi-Zaad Depatment of Electrical and Computer Engineering Queen's University Kingston, ON khz@ece.queensu.ca

Kingston Home
News
Events
Executive
About Kingston

IEEE Canada
IEEE Home
Join IEEE

About Kingston

The Kingston Section covers an area from Belleville in the East to Brockville and Cornwall in the west and north to Smith Falls.

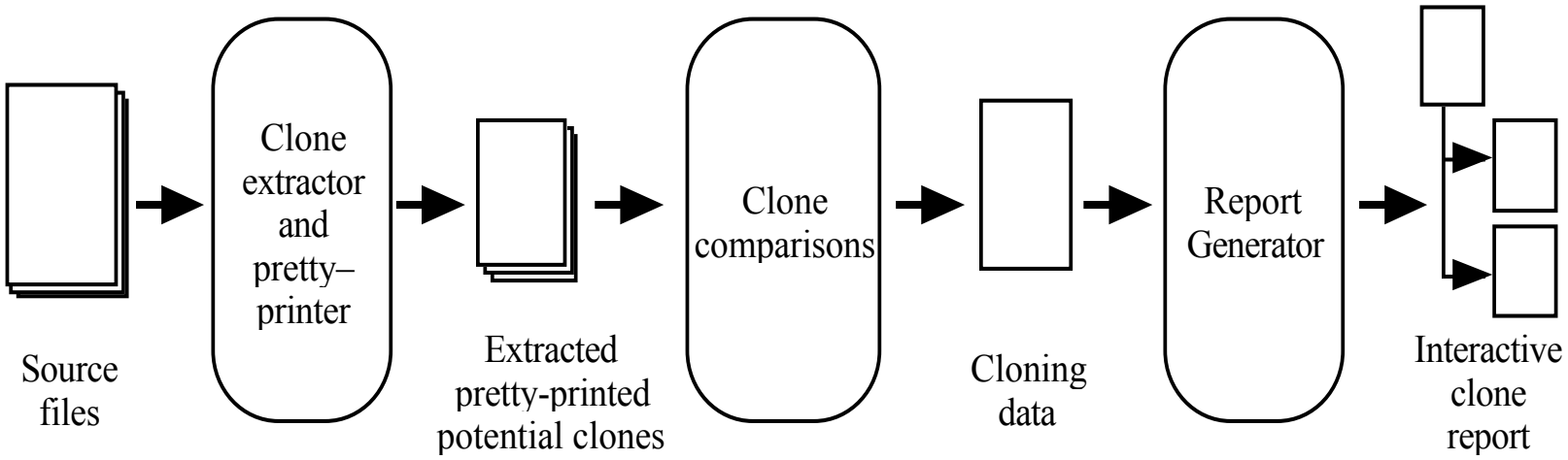
The city of Kingston, is located at the head of the St. Lawrence River, with views of the Lake Ontario and the Rideau Canal.

Last updated, February 17, 2004. Suggestions for improvement may be sent to thomas.dean@ece.queensu.ca

Mixing Lexical and Structural

- We mix the two levels in our approach
- Selection of potential clones is based on partial parses (island grammars)
- Clone identification is lexical
- Not semantic clones
  - **Looking for cut/paste/modify clones**

System Structure



Potential Clone Extraction

- Island Grammars[van Deursen99][Synytskyy03]
- Separate the token stream into interesting (islands) and uninteresting (water) segments
- Islands in a sea of water
  - **Pick out interesting elements in an otherwise uninteresting sequence of tokens**
- Nesting
  - **Islands can have lakes which can have smaller islands**

Islands

```
define program
  [repeat content]
end define

define content
      [interesting_element]
    | [uninteresting element]
end define
```

Islands

```
define interesting_element
        [table_element]
    |   [form_element]
end define

define uninteresting_element
     [token_or_key]
end define
```

Islands and Formatting

- Islands identify potential clones (tables & forms)
  - **extracted to individual files**
- Formatted to make best use of lexical tools
  - **Important elements get their own lines**
  - **Unimportant elements share lines**

```
<table
border=1
id="innerTable1"
>
<tr>
```

```
<td>
content of table cell
</td>
</tr>
</table>
```

Lexical Comparison

- Unix diff
  - **Ignore white space (indentation)**
  - **Ignore addition and deletion of blank lines**
- Threshold based on # of lines that change
  - **Number of unique lines / Total # of Lines**
  - **Experimentally set (tried up to 70% unique code)**
  - **30%**
- Diff finds longest common subsequences
  - **Some common lines are counted as different**

Lexical Comparison

- Special Cases
  - **Identical clones (0% difference)**
  - **Only one file has unique lines (addition/deletion)**
    - Experimentally found to be nesting, so discarded
- Sub clones
  - **Children of clones are clones?**
    - In identical clones, uninteresting
    - In near miss clones, interesting

Advantages

- No need for full language analysis
  - **Complete parses are not necessary**
  - **Semantic Analysis is not necessary**
  - **Less overhead**
- Islands are only used to identify potential clones
- Comparisons are Lexical
  - **Simplicity and speed**

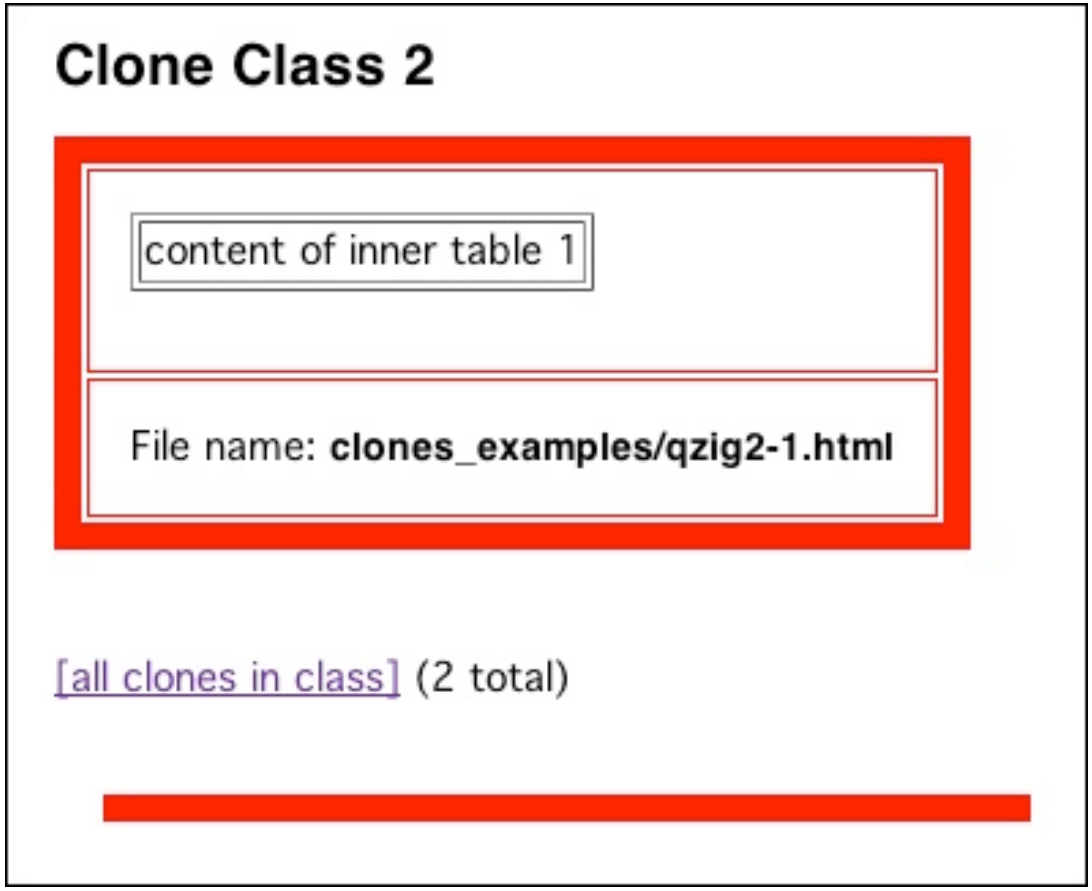
Optimization

- Many, many comparisons of files
  - **Run diff, count lines, compute ratio**
  - **Diff is primary cost >> reduce # of times diff is run**
- Size Restrictions
  - **Files must be comparable in size (max 200%)**
- Group matches into clone classes
  - **Distinguished element of class is the exemplar**
  - **Only compare to the exemplar, not to other members of the class**

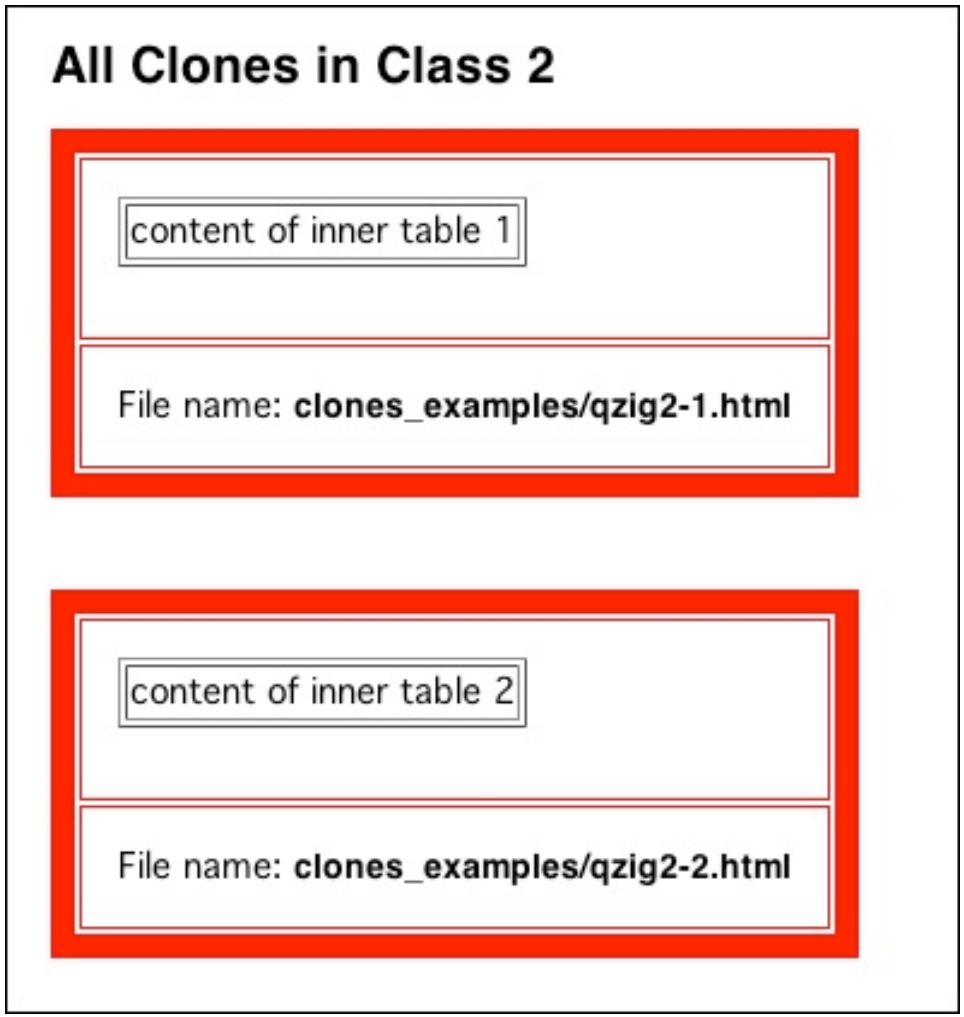
Reporting

- Difficult to execute clones in conventional programming languages in isolation from the whole program
  - **Execution results difficult to visualize**
  - **Must examine code view of clones**
- HTML elements are “executable” by the browser
  - **Can visualize directly**
- Clone Report
  - **One master page shows the exemplars**
  - **Each exemplar is linked to a page that shows all members of the clone class**

Reporting (Exemplar Page)



Reporting (Clone Class pages)



Case Study: Results

	SGPS	TXL
Lines of Code	10378	9436
Potential Clones	686	170
Comparisons	2543	883
Clone Classes	37	24
Time	5 mins	3.75 mins

Case Study: Stats

- 5.4% of potential clones unique in SGPS
- 23.5% of potential clones unique in TXL
- Does not represent true cloning rates since only islands considered as potential clones
  - **Water not included in statistics**
  - **Water could have clones, but we don't look for them (i.e. cloned text)**

Case Study: Types of Clones

- Look and Feel
  - **Large items such as navigation bars (SPGS)**
- Layout
  - **Smaller items, multiple items on a single page (Txl download links)**
- Whole Page
  - **Small clone classes, not a lot of page cloning in our sample set**

Future Work

- What is a near miss clone?
  - **min 70% code share (max 30% unique code) is an initial approximation**
  - **Not a good long term answer**
  - **Tunable parameter?**
- Any near miss clone metric is, to some extent, subjective

Future Work

- Clone Resolution
  - **Synytskyy, Cordy, Dean (WSE 2003)**
  - **Dynamic generation of clones using server side or client side scripting**
    - Parameterize changes
  - **Interactive authoring interface to fine tune clone detection and choose between resolution methods**

Questions?