Thing Compiler

Jonathan Bachrach

EECS UC Berkeley

August 27, 2015
Software Compiler

- goes from program to machine code
- involves many passes
  - lexing + parsing
  - optimization
  - backend
- shape
- color
- density
- joints
- ...
- design is input
- machine + human are targets
- passes
  - rationalization – design -> parts
  - seams + joinery – parts -> parts
  - layout – parts -> polygons
  - instructions – parts -> doc/viz
Mesh Example

- break mesh into developable submeshes
- add joinery
- label and layout
- produce order and animation to show construction
Rationalization

- Simplify input into cheap and easy to manufacture parts
- Example
  - origami
  - panelization
  - slicing
  - puzzle pieces
- provide connection
- joinery system
  - glue
  - zip ties
  - shoe laces
  - rivets
  - puzzle joinery
- joints
  - (living) hinge
  - bracket
  - perforated edge
Nesting

- pack parts to maximize of stock use
- don’t have a robotic assembler
- human is another machine target
- eventually robot will do assembly
- several ways to communicate assembly instructions
Domain Specific Compilers

- supporting all possible meshes is hard
- limiting set of shapes
  - makes writing compiler easier
  - makes support specific fabrication machine easier
  - correct by construction
Material System

- material – sheet steel, leather, plaster
- fabrication – cutting, casting, milling
- joinery – jointing, glueing, stacking
- rationalization – panelization, slicing, subdividing
limited design
- overlapping thin plates
- at least one orthogonal angles
alternative is to union of plates meshes
Space of Compilers
- plate movie
- otherfab examples
mesh
- mesh = list of triangles
- triangle = three points
- points = three coordinates
- obeys winding order

polygon
- polygon = tree of polylines
- polyline = list of points
### Intermediate Representations

<table>
<thead>
<tr>
<th>Name</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>rationalization</td>
<td>mesh</td>
<td>polygons</td>
</tr>
<tr>
<td>joinery</td>
<td>polygons</td>
<td>polygons</td>
</tr>
<tr>
<td>layout</td>
<td>polygons</td>
<td>pdf</td>
</tr>
<tr>
<td>instructions</td>
<td>mesh + polygons</td>
<td>pdf</td>
</tr>
</tbody>
</table>

**Diagram:**

1. **high-level description of a design**
2. **geometric rationalization** (pass A)
3. **joints and seams** (pass B)
4. **layout automation** (pass C)
5. **instructions for assembly** (pass D)
6. **low-level fabrication data**
<table>
<thead>
<tr>
<th>name</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rat</td>
<td>break into 3d parts</td>
</tr>
<tr>
<td>seams</td>
<td>puzzle joinery</td>
</tr>
<tr>
<td>labeling</td>
<td>adjacency</td>
</tr>
<tr>
<td>instructions</td>
<td>assembly ordering</td>
</tr>
<tr>
<td>name</td>
<td>description</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>rat</td>
<td>create slices</td>
</tr>
<tr>
<td>seams</td>
<td>slots</td>
</tr>
<tr>
<td>labeling</td>
<td>xy order</td>
</tr>
<tr>
<td>instructions</td>
<td>assembly ordering</td>
</tr>
<tr>
<td>name</td>
<td>description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>rat</td>
<td>mesh to flat parts</td>
</tr>
<tr>
<td>seams</td>
<td>bends and attachments</td>
</tr>
<tr>
<td>labeling</td>
<td>neighbors</td>
</tr>
<tr>
<td>instructions</td>
<td>assembly ordering</td>
</tr>
<tr>
<td>name</td>
<td>description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>rat</td>
<td>break into plates</td>
</tr>
<tr>
<td>seams</td>
<td>dove tail joinery</td>
</tr>
<tr>
<td>labeling</td>
<td>neighbors</td>
</tr>
<tr>
<td>instructions</td>
<td>assembly ordering</td>
</tr>
<tr>
<td>name</td>
<td>description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>rat</td>
<td>break into edges</td>
</tr>
<tr>
<td>seams</td>
<td>crimp and holes + angles</td>
</tr>
<tr>
<td>labeling</td>
<td>neighbors</td>
</tr>
<tr>
<td>instructions</td>
<td>assembly ordering + crimp instructions</td>
</tr>
</tbody>
</table>
LLVM is compiler toolkit for software compilers
- promotes reuse
- allows writing transformation passes