Invasion Percolation: Neighbors

Program Design

Invasion Percolation: Neighbors

Copyright © Software Carpentry 2010
This work is licensed under the Creative Commons Attribution License
See http://software-carpentry.org/license.html for more information.
Need to find neighbors of marked cells

<table>
<thead>
<tr>
<th>5</th>
<th>3</th>
<th>7</th>
<th>2</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>9</td>
<td>3</td>
<td>9</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>8</td>
<td>8</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>8</td>
<td>5</td>
<td>5</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>8</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>1</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>6</td>
<td>5</td>
<td>1</td>
</tr>
</tbody>
</table>
Need to find neighbors of marked cells

...which are marked with -1
Blue cell is a neighbor if any of the green cells have already been filled.
Those cells coordinates are the center cell's $\pm 1$ to either $x$ or $y$.
We're assuming diagonal cells don't count
We're assuming diagonal cells don't count

Need to check the science...
# Is a cell a candidate for filling?
# Version 1: has bugs!

for x in range(N):
    for y in range(N):
        if is_filled(grid, x-1, y) \ 
        or is_filled(grid, x+1, y) \ 
        or is_filled(grid, x, y-1) \ 
        or is_filled(grid, x, y+1):
            ...cell (x, y) is a candidate...
What do we do at the edges?
[0-1] == -1 wraps around to the other edge
\[(N-1)+1\] == N is an out-of-bounds exception
# Is a cell a candidate for filling?
# Version 2: long-winded
for x in range(N):
  for y in range(N):
    if x > 0:
      if is_filled(grid, x-1, y):
        ...cell (x, y) is a candidate...
        ...repeat for the other three cases...
Is a cell a candidate for filling?

Version 2: long-winded

for x in range(N):
    for y in range(N):
        if x > 0:
            if is_filled(grid, x-1, y):
                ...cell (x, y) is a candidate...
        ...repeat for the other three cases...

"Code that is repeated in two or more places will eventually be wrong in at least one."
# Is a cell a candidate for filling?
# Version 3: good enough for production
for x in range(N):
    for y in range(N):
        if (x > 0) and is_filled(x-1, y)
or (x < N-1) and is_filled(x+1, y)
or (y > 0) and is_filled(x, y-1)
or (y < N-1) and is_filled(x, y+1):
            ...cell (x, y) is a candidate...
if \((x > 0)\) and is\_filled(x-1, y)

Not on the left edge
if (x > 0) and is_filled(x-1, y)

Not on the left edge

Neighbor is already filled
Short-circuit evaluation

```python
if (x > 0) and is_filled(x-1, y)
```

- Not on the left edge
- Neighbor is already filled

Program Design

Invasion Percolation

Neighbors
if sanity_check and some_other_test:

Make sure the second test won't blow up

Only execute if it's safe to do so

Don't try second part if first part is False because the answer is already known
created by

Greg Wilson

May 2010

Copyright © Software Carpentry 2010
This work is licensed under the Creative Commons Attribution License
See http://software-carpentry.org/license.html for more information.