Announcements (2/27/09)

- Homework 6 due now.
- Homework 7 assigned.
- Reading for this lecture: Chapter 8.

Making Connections
You have a set of nodes (numbered 1-9) on a network. You are given a sequence of pairwise connections between them:

3-5
4-2
1-6
5-7
4-8
3-7

Q: Are nodes 2 and 4 (indirectly) connected?
Q: How about nodes 3 and 8?
Q: Are any of the paired connections redundant due to indirect connections?
Q: How many sub-networks do you have?

Making Connections
Answering these questions is much easier if we create disjoint sets of nodes that are connected:

Start: \{1\} \{2\} \{3\} \{4\} \{5\} \{6\} \{7\} \{8\} \{9\}
3-5
4-2
1-6
5-7
4-8
3-7

Q: Are nodes 2 and 4 (indirectly) connected?
Q: How about nodes 3 and 8?
Q: Are any of the paired connections redundant due to indirect connections?
Q: How many sub-networks do you have?
Applications of Disjoint Sets

Maintaining disjoint sets in this manner arises in a number of areas, including:

– Networks
– Transistor interconnects
– Compilers
– Image segmentation
– Building mazes (this lecture)
– Graph problems
  • Minimum Spanning Trees (upcoming topic in this class)

Disjoint Set ADT

• Data: set of pairwise **disjoint sets**.
• Required operations
  – **Union** – merge two sets to create their union
  – **Find** – determine which set an item appears in
• A common operation sequence:
  – Connect two elements if not already connected:
    \[
    \text{if (Find}(x) \neq \text{Find}(y) \text{) then Union}(x, y)
    \]

Disjoint Sets and Naming

• Maintain a set of pairwise disjoint sets.
  – \{3,5,7\}, \{4,2,8\}, \{9\}, \{1,6\}
• Each set has a unique name: one of its members (for convenience)
  – \{3,5,7\}, \{4,2,8\}, \{9\}, \{1,6\}

Union

• **Union**(x,y) – take the union of two sets named x and y
  – \{3,5,7\}, \{4,2,8\}, \{9\}, \{1,6\}
  – Union(5,1)
    \{3,5,7,1,6\}, \{4,2,8\}, \{9\}
Find

- Find(x) – return the name of the set containing x.
  - {3, 5, 7, 1, 6}, {4, 2, 8}, {9},
  - Find(1) = 5
  - Find(4) = 8

Example

- \{1, 2, 7, 8, 9, 13, 19\}
- \{3\}
- \{4\}
- \{5\}
- \{6\}
- \{10\}
- \{11, 17\}
- \{12\}
- \{14, 20, 26, 27\}
- \{15, 16, 21\}
- \{22, 23, 24, 29, 39, 32\}
- \{33, 34, 35, 36\}

\[\text{Find(8) = 7}\]
\[\text{Find(14) = 20}\]
\[\text{Union(7, 20)}\]

Nifty Application: Building Mazes

Idea: Build a random maze by erasing walls.

Building Mazes

- Pick Start and End
Building Mazes

- Repeatedly pick random walls to delete.

Desired Properties

- None of the boundary is deleted (except at “start” and “end”).
- Every cell is reachable from every other cell.
- There are no cycles – no cell can reach itself by a path unless it retraces some part of the path.
A Hidden Tree

Number the Cells

We start with disjoint sets \( S = \{ \{1\}, \{2\}, \{3\}, \{4\}, \ldots \{36\} \} \).
We have all possible walls between neighbors \( W = \{ (1,2), (1,7), (2,8), (2,3), \ldots \} \) 60 walls total.

<table>
<thead>
<tr>
<th>Start</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>End</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Idea: Union-find operations will be done on cells.

Maze Building with Disjoint Union/Find

Algorithm sketch:
1. Choose wall at random.
   → \textit{Boundary walls are not in wall list, so left alone}
2. Erase wall if the neighbors are in disjoint sets.
   → \textit{Avoids cycles}
3. Take union of those sets.
4. Go to 1, iterate until there is only one set.
   → \textit{Every cell reachable from every other cell.}

Pseudocode

\begin{itemize}
\item \( S \) = set of sets of connected cells
  \hspace{1cm} – Initialize to \( \{\{1\}, \{2\}, \ldots, \{n\}\} \)
\item \( W \) = set of walls
  \hspace{1cm} – Initialize to set of all walls \( \{(1,2),(1,7),\ldots\} \)
\item \( \text{Maze} \) = set of walls in maze (initially empty)
\end{itemize}

\begin{verbatim}
While there is more than one set in S
    Pick a random non-boundary wall \((x,y)\) and remove from W
    \( u = \text{Find}(x); \)
    \( v = \text{Find}(y); \)
    if \( u \neq v \) then
        \( \text{Union}(u,v) \)
    else
        Add wall \((x,y)\) to Maze
Add remaining members of W to Maze
\end{verbatim}