Largest Puzzle in Vermont

www.vermontcornmaze.com (2006 design)
What is a maze?

A maze is a network of passages connected by junctions, with an entrance and a goal.
What is a maze?

This maze can be represented using a graph, where

- a green dot represents the entrance,
What is a maze?

This maze can be represented using a graph, where

- a **green dot** represents the entrance,
- **blue dots** represent the junctions,
- **red dots** represent the dead ends,
- a **gold dot** represents the goal.
What is a maze?

This maze can be represented using a graph, where

- a green dot represents the entrance,
- blue dots represent the junctions,
- red dots represent the dead ends,
What is a maze?

This maze can be represented using a graph, where
- a green dot represents the entrance,
- blue dots represent the junctions,
- red dots represent the dead ends,
- a gold dot represents the goal, and
What is a maze?

This maze can be represented using a graph, where
- a green dot represents the entrance,
- blue dots represent the junctions,
- red dots represent the dead ends,
- a gold dot represents the goal, and
- brown lines connect adjacent dots.
How does one “thread” a maze?
How does one “thread” a maze?

Guess a solution
How does one “thread” a maze?

Or, use an *algorithm*:
How does one “thread” a maze?

Or, use an algorithm:

*a sequence of instructions, steps, or rules, designed to solve a class of problems. An algorithm is the heart of every computer program.*
How does one “thread” a maze?

Or, use an *algorithm*:

*a sequence of instructions, steps, or rules, designed to solve a class of problems. An algorithm is the heart of every computer program.*

A popular algorithm for solving mazes is *wall following*:
How does one “thread” a maze?

Or, use an algorithm:

a sequence of instructions, steps, or rules, designed to solve a class of problems. An algorithm is the heart of every computer program.

A popular algorithm for solving mazes is wall following:

As you enter the maze, place your left or right hand on the corresponding wall. Then explore the maze, while continuously touching that wall: don’t let go.
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand
Wall-following with the left hand
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand

Wall-following doesn't always work!
Wall-following with the left hand
Wall-following with the left hand

Wall-following doesn’t always work!
Trémaux’s Algorithm

The maze walker will draw symbols ($N$ or $X$) at the ends of paths as they are visited.

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Marching forward into a new junction:</td>
<td><img src="image" alt="Diagram" /></td>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>
Trémaux’s Algorithm

The maze walker will draw symbols (\(N\) or \(X\)) at the ends of paths as they are visited.

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Marching forward into a new junction:</td>
<td><img src="https://via.placeholder.com/150" alt="Diagram" /></td>
<td><img src="https://via.placeholder.com/150" alt="Diagram" /></td>
</tr>
<tr>
<td>● Mark the exit of the current path with an (X).</td>
<td><img src="https://via.placeholder.com/150" alt="Diagram" /></td>
<td><img src="https://via.placeholder.com/150" alt="Diagram" /></td>
</tr>
</tbody>
</table>
Trémaux’s Algorithm

The maze walker will draw symbols (\(N\) or \(X\)) at the ends of paths as they are visited.

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Marching <strong>forward</strong> into a <strong>new</strong> junction:</td>
<td><img src="image1.png" alt="Before Diagram" /></td>
<td><img src="image2.png" alt="After Diagram" /></td>
</tr>
<tr>
<td>- Mark the exit of the current path with an (X).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Select a <strong>new</strong> path.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Trémaux’s Algorithm

The maze walker will draw symbols (N or X) at the ends of paths as they are visited.

<table>
<thead>
<tr>
<th>SITUATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Marching <strong>forward</strong> into a <strong>new</strong> junction:</td>
</tr>
<tr>
<td>- Mark the exit of the current path with an <strong>X</strong>.</td>
</tr>
<tr>
<td>- Select a <strong>new</strong> path.</td>
</tr>
<tr>
<td>- Mark the entrance of the <strong>new</strong> path with an <strong>N</strong>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BEFORE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image-url" alt="Diagram" /></td>
</tr>
</tbody>
</table>
Trémaux’s Algorithm

The maze walker will draw symbols ($N$ or $X$) at the ends of paths as they are visited.

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Marching <strong>forward</strong> into a <strong>new</strong> junction:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Mark the exit of the current path with an $X$.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Select a <strong>new</strong> path.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Mark the entrance of the <strong>new</strong> path with an $N$.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- March forward.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2. Marching **forward** into an **old** junction:

<table>
<thead>
<tr>
<th><strong>SITUATION</strong></th>
<th><strong>BEFORE</strong></th>
<th><strong>AFTER</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image" alt="Diagram" /></td>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>

© 2013 Computational Thinking with Mazes
Trémaux’s Algorithm

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. Marching <strong>forward</strong> into an old junction:</td>
<td><img src="image" alt="Diagram" /></td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td>• Mark the exit of the current path with an <strong>N</strong>.</td>
<td><img src="image" alt="Diagram" /></td>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>
Trémaux’s Algorithm

2. Marching forward into an old junction:
   - Mark the exit of the current path with an $N$.
   - Turn around.
   - March backward along the current path.
### Trémaux’s Algorithm

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>3. Marching <strong>forward</strong> into a dead end:</td>
<td><img src="image1.png" alt="Diagram 1" /></td>
<td><img src="image2.png" alt="Diagram 2" /></td>
</tr>
</tbody>
</table>
### Trémaux’s Algorithm

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>3. Marching forward into a dead end:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Turn around.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• March backward.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Trémaux’s Algorithm

4. Marching **backward** into an **old** junction with *no* unlabeled paths

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Marching backward into an old junction with <em>no</em> unlabeled paths</td>
<td><img src="image" alt="Diagram" /></td>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>
Trémaux’s Algorithm

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Marching <strong>backward</strong> into an old junction with <em>no</em> unlabeled paths</td>
<td><img src="" alt="Diagram" /></td>
<td><img src="" alt="Diagram" /></td>
</tr>
<tr>
<td>Select the path with the $X$.</td>
<td><img src="" alt="Diagram" /></td>
<td><img src="" alt="Diagram" /></td>
</tr>
</tbody>
</table>
4. Marching **backward** into an old junction with *no* unlabeled paths

- Select the path with the $X$.
- Continue marching backward.
Trémaux’s Algorithm

<table>
<thead>
<tr>
<th>SITUATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. Marching <strong>backward</strong> into an old junction with <em>some</em> unlabeled paths</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BEFORE</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Diagram Before]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Diagram After]</td>
</tr>
</tbody>
</table>
Trémaux’s Algorithm

<table>
<thead>
<tr>
<th>SITUATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. Marching <strong>backward</strong> into an old junction with <em>some</em> unlabeled paths</td>
</tr>
<tr>
<td>Select an unlabeled path.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BEFORE</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://via.placeholder.com/150" alt="Diagram" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://via.placeholder.com/150" alt="Diagram" /></td>
</tr>
</tbody>
</table>
Trémaux’s Algorithm

5. Marching *backward* into an old junction with *some* unlabeled paths
   - Select an unlabeled path.
   - Mark the entrance of the new path with an $N$.

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. Marching <em>backward</em> into an old junction with <em>some</em> unlabeled paths</td>
<td><img src="image" alt="Diagram of the before scenario" /></td>
<td><img src="image" alt="Diagram of the after scenario" /></td>
</tr>
</tbody>
</table>

© 2013 Computational Thinking with Mazes Code Camp, Summer 2013
### SITUATION

5. Marching **backward** into an *old* junction with *some* unlabeled paths
   - Select an unlabeled path.
   - Mark the entrance of the *new* path with an **N**.
   - March **forward**.

<table>
<thead>
<tr>
<th>SITUATION</th>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. Marching backward into an old junction with some unlabeled paths</td>
<td>![Diagram Before]</td>
<td>![Diagram After]</td>
</tr>
</tbody>
</table>

© 2013 Computational Thinking with Mazes Code Camp, Summer 2013
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an \( \times \).
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.
- Select a new path.
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.
- Select a new path.
- Mark the new path entrance with an $N$. 
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.
- Select a new path.
- Mark the new path entrance with an $N$.
- March forward.
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$. 

© 2013 Computational Thinking with Mazes
Code Camp, Summer 2013
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:

- Mark path exit with an $X$.
- Select a new path.
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.
- Select a new path.
- Mark the new path entrance with an $N$.
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.
- Select a new path.
- Mark the new path entrance with an $N$.
- March forward.
Trémaux’s algorithm: left-bearing mouse

Mark path exit with an $X$. Select a new path. March forward. Mark the new path entrance with an $N$. March forward into a new junction:
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an X.
- Select a new path.
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:

- Mark path exit with an $X$.
- Select a new path.
- Mark the new path entrance with an $N$.
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:

- Mark path exit with an \( X \).
- Select a new path.
- Mark the new path entrance with an \( N \).
- March forward.
Trémaux’s algorithm: left-bearing mouse

Marching forward into a dead end:
Trémaux’s algorithm: left-bearing mouse

Marching forward into a dead end:
- Turn around,
- March backward.
Trémaux’s algorithm: left-bearing mouse

Marching **backward** into an **old** junction with unlabeled paths:
Trémaux’s algorithm: left-bearing mouse

Marching *backward* into an *old* junction with unlabeled paths:

- Select a *new* path.
Trémaux’s algorithm: left-bearing mouse

Marching **backward** into an **old** junction with unlabeled paths:
- Select a **new** path.
- Mark the **new** path entrance with an **N**.
Trémaux’s algorithm: left-bearing mouse

Marching **backward** into an **old** junction with unlabeled paths:

- Select a **new** path.
- Mark the **new** path entrance with an **N**.
- March **forward**.
Trémaux’s algorithm: left-bearing mouse

Marching *forward* into a *new* junction:
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.

© 2013 Computational Thinking with Mazes, Code Camp, Summer 2013
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.
- Select a new path.
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.
- Select a new path.
- Mark the new path entrance with an $N$. 
Trémaux’s algorithm: left-bearing mouse

Marching forward into a new junction:
- Mark path exit with an $X$.
- Select a new path.
- Mark the new path entrance with an $N$.
- March forward.
Trémaux’s algorithm: left-bearing mouse

Marching forward into an old junction:

© 2013 Computational Thinking with Mazes Code Camp, Summer 2013 12 / 13
Trémaux’s algorithm: left-bearing mouse

Marching forward into an old junction:
- Mark path exit with an $N$!
Trémaux’s algorithm: left-bearing mouse

Marching **forward** into an **old** junction:
- Mark path exit with an **$N$**!
- Turn around.
- March **backward**.

© 2013 Computational Thinking with Mazes
Trémaux’s algorithm: left-bearing mouse

Marching backward into an old junction with unlabeled paths:
Trémaux’s algorithm: left-bearing mouse

Marching backward into an old junction with unlabeled paths:

- Select a new path.
Trémaux’s algorithm: left-bearing mouse

Marching backward into an old junction with unlabeled paths:

- Select a new path.
- Mark the new path entrance with an \( N \).
Trémaux’s algorithm: left-bearing mouse

Marching **backward** into an **old** junction with unlabeled paths:
- Select a **new** path.
- Mark the **new** path entrance with an **N**.
- March **forward**.
Trémaux’s algorithm: left-bearing mouse

Arriving at goal:

Shout “eureka”!

To get back to the entrance, follow the Xs.
Trémaux’s algorithm: left-bearing mouse

Arriving at goal:
- Shout “eureka”!
Trémaux’s algorithm: left-bearing mouse

Arriving at goal:

- Shout “eureka”!
- To get back to the entrance, follow the Xs.
Practice with Mini-Mazes