Lecture 25: Sprouts & Brussel Sprouts
Outline for Today

● Final Project Presentation Schedule
● Last Time: Robot Motion Planning, Minkowski Sums, etc.
● Paper & Pencil Game of Sprouts
● Computer Analysis of Sprouts
● Brussel Sprouts
● Hybrid Sprouts & Brussel Sprouts
● Next Time: Final Project Presentations!
Final Project Presentation Schedule

8 min for individual + 1 min for questions, 16 min for team of 2 + 2 min for questions

2:00
2:18
2:27
2:36
2:54
3:03
3:12
3:30
3:39
3:48 done!
Outline for Today

- Final Project Presentation Schedule
- Last Time: Robot Motion Planning, Minkowski Sums, etc.
- Paper & Pencil Game of Sprouts
- Computer Analysis of Sprouts
- Brussel Sprouts
- Hybrid Sprouts & Brussel Sprouts
- Next Time: Final Project Presentations!
Robot Degree of Freedom (DOF)

2D w/ Translation only → 2 DOF

2D w/ Translation & Rotation → 3 DOF

Computational Geometry Algorithms and Applications,
de Berg, Cheong, van Kreveld and Overmars, Chapter 13
Configuration Space

- The dimensions of configuration space match the DOF of the robot.
- Usually configuration space is higher dimensional than the environment/workspace.
- It is often useful to construct, visualize, and even solve the problem in “configuration space”.

2D w/ translation only → 2 DOF

Reference point

Work space

Configuration space
Motion Planning Graph - Analysis

- Size of Trapezoid Map
  → $O(n)$

- Build Trapezoid Map
  → $O(n \log n)$

- Locate start/end trapezoid
  → $O(\log n)$

- Breadth first search
  → $O(n)$
Searching Configuration Space

- Dimensionality becomes infeasible to construct & exhaustively search
- Randomized search is necessary

Outline for Today

● Final Project Presentation Schedule
● Last Time: Robot Motion Planning, Minkowski Sums, etc.
● Paper & Pencil Game of Sprouts
● Computer Analysis of Sprouts
● Brussel Sprouts
● Hybrid Sprouts & Brussel Sprouts
● Next Time: Final Project Presentations!
Sprouts Game Rules

- Draw $n$ spots
- Players take turns:
  - Draw a line joining two spots, or a single spot to itself.
  - The line must not cross another line or pass through another spot.
  - Draw a spot on the new line.
  - No more than three lines can emerge from any spot.
- Normal Winning Condition: Winner is last person to make a move
- Misère Winning Condition: Winner is first person who cannot make a move
Sprouts Analysis

- Starting with $n$ dots
- What’s the maximum number of turns?
  What’s the maximum number of lines drawn?
  What’s the maximum number of new points added?
Sprouts Analysis

- Starting with \( n \) dots
- What’s the minimum number of turns?
Definition: Planar Graph

- Can be drawn in 2D without any edges crossing
Sprout Game Tree

Even for just 2 starting spots, the full tree of moves is surprisingly large!

"Computer Analysis of Sprouts”, Applegate, Jacobson, & Sleator, 1991
Sprout Game Tree
Normal Play

Maximum # moves = 5

If the game ends in 5 moves, Player 1 makes the last move and wins

Sprout Game Tree - Normal Play

If Player 2 can separate and isolate the 2 final dots, ending in 4 moves, they win!

Sprout Game Tree - Misère Play

Flip the condition: Player who makes last move loses

Does this mean Player 1 can guarantee a win?

Sprout Game Tree - Misère Play

Nope! Player 2 is still the winner with perfect play

Outline for Today

- Final Project Presentation Schedule
- Last Time: Robot Motion Planning, Minkowski Sums, etc.
- Paper & Pencil Game of Sprouts
- Computer Analysis of Sprouts
- Brussel Sprouts
- Hybrid Sprouts & Brussel Sprouts
- Next Time: Final Project Presentations!
A “1” means the first player to move has a winning strategy, a “2” means the second player has a winning strategy, and an asterisk indicates a new result obtained by our program.

The $n$-spot Sprouts positions evaluated so far fall into a remarkably simple pattern, characterized by the following conjecture:

**Sprouts conjecture.** The first player has a winning strategy in $n$-spot Sprouts if and only if $n$ is 3, 4, or 5 modulo 6. **STILL HOLDS!**

The data for misère Sprouts fit a similar pattern.

**Misère sprouts conjecture.** The first player has a winning strategy in $n$-spot misère Sprouts if and only if $n$ is 0 or 1 modulo 5. **LATER DISPROVEN**

<table>
<thead>
<tr>
<th>Number of Spots</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal play</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2*</td>
<td>2*</td>
<td>1*</td>
<td>1*</td>
<td>1*</td>
</tr>
<tr>
<td>misère play</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1*</td>
<td>1*</td>
<td>2*</td>
<td>2*</td>
<td>2*</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Definition: Graph Isomorphism

Two graphs are isomorphic if there is a bijection between the labels of the graphs, and an edge exists between a pair of vertices in one graph if and only if an edge exists between the corresponding vertices in the other graph.

![Graph Isomorphism Example](https://en.wikipedia.org/wiki/Graph_isomorphism)

An isomorphism between G and H:
- \( f(a) = 1 \)
- \( f(b) = 6 \)
- \( f(c) = 8 \)
- \( f(d) = 3 \)
- \( f(g) = 5 \)
- \( f(h) = 2 \)
- \( f(i) = 4 \)
- \( f(j) = 7 \)
Sprouts Analysis Definitions

- Each spot is given a unique name.
- The curves of the sprout graph divide the plane into *regions*.
- The loop of spots and curves surrounding a region is called a *boundary*.
- The sprout graph can be encoded in set representation:
  \[
  \{ \{ (abcdhfcb) (ijkj) (l) \} \{ (cfed) \} \{ (degh) \} \{ (efhg) \} \}
  \]
Sprouts Analysis Move Definitions

A **two-boundary move** joins spot $x$ on boundary $B_1$ to spot $y$ on boundary $B_2$ and adds spot $z$. Boundaries $B_1$ and $B_2$ are now joined. No new regions are formed.

A **one-boundary move** connects spots on the same boundary of region $R$. Region $R$ is split into two subregions $R_1$ and $R_2$. All boundaries of and within $R$ must be designated for $R_1$, or $R_2$.

*Note: $2^{k-1}$ ways to group $k$ boundaries*
Sprouts Analysis Representation Simplification

The String representation can be simplified/compressed:

- Spots of degree 3 can be thrown away
- Regions with fewer than 2 lives can be thrown away

```
abdhfcb•ijkj•l••cfed••degh••efhg••
```

```
abb•ijkj•l••••g•••g••
```
Sprouts Analysis Representation Simplification

The String representation can be simplified/compressed:

- Spots of degree 3 can be thrown away
- Regions with fewer than 2 lives can be thrown away
- Boundaries with no live spots can be thrown away
- Spots of degree zero or one do not appear on more than one boundary, so they don’t need unique names, label them 0 and 1.
- If a degree two spot has no live spots between its two occurrences on one boundary, it only needs to be listed once.
function eval(P)
    \( S \leftarrow \text{successors}(P) \)
    if \( S \) is empty then return “L”
    for each position \( P' \in S \) do
        if \( P' \) is in the hash table return “W”
    for each position \( P' \in S \) do
        if eval(\( P' \)) is “L” then return “W”
    put \( P \) into the hash table;
    return “L”
end eval
Sprouts Analysis Implementation

- Lexicographically sort the boundaries & regions to remove (significantly reduce) duplicate/isomorphic graphs.
- Use a hash table to store compressed set representation of all sprout graphs that we have previously examined & labeled “W” or “L”.

Implementation notes with 1990 hardware: *may be different now*

- Memory was a more significant resource limitation than CPU/time.
- There are 10X more “W” configurations than “L” configurations.
- Therefore, only store “L” configurations in the hash table & recompute successors when needed.
### Sprouts Perfect Play Winner - Normal Play

Analysis now complete through $n = 44$ (2011)

Conjecture: Player 1 wins if # spots $\% 6 = 3, 4, \text{or} 5$

<table>
<thead>
<tr>
<th>Number of spots</th>
<th>Value of game</th>
<th>Cpu seconds (on a DEC 5000)</th>
<th># of positions in hash table</th>
<th>size of hash table (in bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>L</td>
<td>$&lt; 0.1$</td>
<td>1</td>
<td>101</td>
</tr>
<tr>
<td>2</td>
<td>L</td>
<td>$&lt; 0.1$</td>
<td>4</td>
<td>606</td>
</tr>
<tr>
<td>3</td>
<td>W</td>
<td>$&lt; 0.1$</td>
<td>7</td>
<td>606</td>
</tr>
<tr>
<td>4</td>
<td>W</td>
<td>0.2</td>
<td>33</td>
<td>1515</td>
</tr>
<tr>
<td>5</td>
<td>W</td>
<td>1.1</td>
<td>114</td>
<td>2828</td>
</tr>
<tr>
<td>6</td>
<td>L</td>
<td>5.9</td>
<td>338</td>
<td>4070</td>
</tr>
<tr>
<td>7</td>
<td>L</td>
<td>75.8</td>
<td>1843</td>
<td>16794</td>
</tr>
<tr>
<td>8</td>
<td>L</td>
<td>1813.7</td>
<td>24842</td>
<td>264756</td>
</tr>
<tr>
<td>9</td>
<td>W</td>
<td>8.9</td>
<td>24897</td>
<td>264756</td>
</tr>
<tr>
<td>10</td>
<td>W</td>
<td>842.8</td>
<td>33252</td>
<td>354721</td>
</tr>
<tr>
<td>11</td>
<td>W</td>
<td>10107.6</td>
<td>116299</td>
<td>1467576</td>
</tr>
</tbody>
</table>


Almost 3 hours

1.5 MB, max memory for DEC was 8-480MB (after 1991?)
**Original Conjecture was later disproven**

**Current Conjecture:** Player 1 wins when # spots % 6 == 0, 4, or 5

**EXCEPTIONS:** Player 1 wins if # spots == 1 and Player 1 loses if # spots == 4

---

<table>
<thead>
<tr>
<th>Number of spots</th>
<th>Value of game</th>
<th>Cpu seconds (on a DEC 5000)</th>
<th># of positions in hash table</th>
<th>size of hash table (in bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>W</td>
<td>&lt; 0.1</td>
<td>1</td>
<td>202</td>
</tr>
<tr>
<td>2</td>
<td>L</td>
<td>&lt; 0.1</td>
<td>5</td>
<td>303</td>
</tr>
<tr>
<td>3</td>
<td>L</td>
<td>&lt; 0.1</td>
<td>11</td>
<td>606</td>
</tr>
<tr>
<td>4</td>
<td>L</td>
<td>0.1</td>
<td>37</td>
<td>1010</td>
</tr>
<tr>
<td>5</td>
<td>W</td>
<td>1.1</td>
<td>219</td>
<td>2307</td>
</tr>
<tr>
<td>6</td>
<td>W</td>
<td>18.9</td>
<td>1805</td>
<td>15403</td>
</tr>
<tr>
<td>7</td>
<td>L</td>
<td>44.0</td>
<td>4970</td>
<td>43618</td>
</tr>
<tr>
<td>8</td>
<td>L</td>
<td>343.7</td>
<td>23728</td>
<td>202364</td>
</tr>
<tr>
<td>9</td>
<td>L</td>
<td>30579.5</td>
<td>1024629</td>
<td>13417664</td>
</tr>
</tbody>
</table>


Analysis now complete through n = 20 (2011)
Discussion: Interactive Sprouts Implementation

- I couldn’t find a good interactive, sketch-based digital Sprouts game
  *Many versions seem to require Flash, so I don’t know if they were any good*
- What is hard about making a computer/phone implementation of Sprouts?
“Implementation of Sprouts: a graph drawing game”
Cizek & Balko, Graph Drawing 2021

Challenge 1:
Free form input drawings degenerate & make positions confusing over time

Use forces to move each vertex such that no crossing is introduced

“Implementation of Sprouts: a graph drawing game”
Cizek & Balko, Graph Drawing 2021

Challenge 2: Enormous Game Tree

Challenge 3: Drawing a Computer Move

“Implementation of Sprouts: a graph drawing game”
Cizek & Balko, Graph Drawing 2021

“Implementation of Sprouts: a graph drawing game”
Cizek & Balko, Graph Drawing 2021

- Support games up to 20 spots
- Perfect AI on up to 11 spots

https://kam.mff.cuni.cz/~cizek/Sprouts/win_x64/
Outline for Today

- Final Project Presentation Schedule
- Last Time: Robot Motion Planning, Minkowski Sums, etc.
- Paper & Pencil Game of Sprouts
- Computer Analysis of Sprouts
- Brussel Sprouts
- Hybrid Sprouts & Brussel Sprouts
- Next Time: Final Project Presentations!
Brussel Sprouts Variant Game Rules

- Draw $n$ crosses
- Players take turns:
  - Draw a line joining two cross “legs”.
  - Draw a hash mark across the new line (a new “cross” position).
  - No more than four lines can emerge from any cross.

https://en.wikipedia.org/wiki/Sprouts_(game)#/media/File:Brussel_Sprouts_Game.png
Brussel Sprouts Analysis

- We start with $n$ crosses and $4n$ live ‘leg’s.
- Each move adds a line, uses up two legs, and adds two legs.
- *Does it ever end?*
Brussel Sprouts Analysis

- Yes, in fact it does end!
  - Every face contains at least one live leg
  - We are done when no face has more than one live leg
- It must follow Euler’s Characteristic for Planar Graphs:
  - Let \( m \) = \# of moves
  - \# of edges = \( e = 2m \)
  - \# of vertices = \( v = n + m \) (we start with \( n \), and we add one each move)
  - \# of faces = \( 4n \), there is exactly 1 free end inside of each face at the end
  - \( 2 = f - e + v \)
  - \( 2 = 4n - 2m + n + m \)
  - \# of moves = \( m = 5n - 2 \)
Outline for Today

- Final Project Presentation Schedule
- Last Time: Robot Motion Planning, Minkowski Sums, etc.
- Paper & Pencil Game of Sprouts
- Computer Analysis of Sprouts
- Brussel Sprouts
- Hybrid Sprouts & Brussel Sprouts
- Next Time: Final Project Presentations!
Hybrid Sprouts & Brussel Sprouts Game Rules

● Start with a mix of dots and/or crosses (which player’s choice?)
● Draw a line connecting a dot or cross to itself or another dot or cross (player choice) and add a dot or a cross to the line (player choice)
Hybrid Sprouts & Brussel Sprouts Analysis

For $n = 1$
Hybrid Sprouts & Brussel Sprouts Analysis

For \( n = 1 \)

*If we start with one dot… the game has 2 moves*

*If we start with one cross… the game has 2 or 3 moves*
Hybrid Sprouts & Brussel Sprouts Analysis

For n = 2?
Hybrid Sprouts & Brussel Sprouts Analysis

For \( n = 2 \)?

Actually… for \( n > 1 \): Analysis is not completed! (according to Wikipedia)
Outline for Today

- Final Project Presentation Schedule
- Last Time: Robot Motion Planning, Minkowski Sums, etc.
- Paper & Pencil Game of Sprouts
- Computer Analysis of Sprouts
- Brussel Sprouts
- Hybrid Sprouts & Brussel Sprouts
- Next Time: Final Project Presentations!
Final Presentation

- Summarize prior work as necessary
  
  *Can assume peers know algorithms/structures/papers from lecture*

- Be technical:
  
  What were the challenges?
  How did you solve them?

- Live demo / video / lots of images (depends on project)
  
  Use plenty of examples (both of success & failure)

- Teams of 2: Both should present & make it clear who did what

- Use your time wisely! Practice! & Time yourself!

  *I will stop you mid-sentence if you run over*
Well-written Research Paper / Report

- Motivation / Context / Related Work
- Accomplishments / Contributions of this work
- Clear description of algorithm
  - Sufficiently-detailed to allow work to be reproduced
  - Work is theoretically sound (hacks/arbitrary constants discouraged, but must be documented)
- Results
  - Well chosen examples
  - Clear tables/illustrations/visualizations with descriptive captions!
- Conclusions & Potential Future Work
  - Limitations of the method are clearly stated
<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nov 27</td>
<td>Final Project Progress Post #2 due @ 11:59pm</td>
<td>Nov 28</td>
<td>Lecture 24: Robot Motion Planning</td>
</tr>
<tr>
<td></td>
<td><strong>Textbook Reading:</strong></td>
<td></td>
<td><em>CGAA Chapter 13</em></td>
</tr>
<tr>
<td>Dec 5</td>
<td>Lecture 25: Sprouts &amp; Brussels Sprouts</td>
<td>Dec 7</td>
<td>Final Project Written Report due @ 11:59pm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dec 8</td>
<td>Final Project Presentations</td>
</tr>
<tr>
<td>Dec 11-13</td>
<td>Reading days</td>
<td>Dec 14-15</td>
<td>Other RPI Final Exams (no Final Exam for Computational Geometry)</td>
</tr>
<tr>
<td></td>
<td>No classes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dec 18-20</td>
<td>Other RPI Final Exams</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>(no Final Exam for Computationaization Geometry)</em></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>