MINICURS:
INTRODUCCIÓ A LA INTEL·LIGÈNCIA ARTIFICIAL
APLICADA ALS SISTEMES COMPLEXOS

INTRODUCCIÓ A LA INTEL·LIGÈNCIA ARTIFICIAL: ALGORITMES DE CERCA

Professor: Dr. Mario Gutiérrez-Roig
University of Essex, UK

http://ubics.ub.edu
http://ubics.ub.edu/AI_course
Artificial Intelligence
### Thinking Humanly
- “The exciting new effort to make computers think . . . *machines with minds*, in the full and literal sense.” (Haugeland, 1985)
- “[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning . . .” (Bellman, 1978)

### Thinking Rationally
- “The study of mental faculties through the use of computational models.” (Charniak and McDermott, 1985)
- “The study of the computations that make it possible to perceive, reason, and act.” (Winston, 1992)

### Acting Humanly
- “The art of creating machines that perform functions that require intelligence when performed by people.” (Kurzweil, 1990)
- “The study of how to make computers do things at which, at the moment, people are better.” (Rich and Knight, 1991)

### Acting Rationally
- “Computational Intelligence is the study of the design of intelligent agents.” (Poole *et al.*, 1998)
- “AI . . . is concerned with intelligent behavior in artifacts.” (Nilsson, 1998)
Turing Test

- **Natural language processing**: Able to communicate in a human language.
- **Knowledge representation**: Store information in a structured way.
- **Automated reasoning**: Use stored information for answering questions.
- **Machine learning**: Detect and extrapolate patterns. Adapt to new or change of data.
- **Computer vision**: Perceive and detect physical objects.
- **Robotics**: Move around and manipulate objects.
A.I. Timeline

1950
- Turing Test
  Computer scientist Alan Turing proposes a test for machine intelligence. If a machine can trick humans into thinking it is human, then it has intelligence.

1955
- A.I. Born
  Term 'artificial intelligence' is coined by computer scientist John McCarthy to describe "the science and engineering of making intelligent machines."

1961
- UNIMATE
  First industrial robot, Unimate, goes to work at GM replacing humans on the assembly line.

1964
- ELIZA
  Pioneering chatbot developed by Joseph Weizenbaum at MIT holds conversations with humans.

1966
- SHAKEY
  The 'first electronic person' from Stanford, Shakey is a general-purpose mobile robot that reasons about its own actions.

A.I. Winter
- Many false starts and dead-ends leave A.I. out in the cold.

1997
- DEEP BLUE
  Deep Blue, a chess-playing computer from IBM defeats world chess champion Garry Kasparov.

1998
- KISMET
  Cynthia Breazeal at MIT introduces Kismet, an emotionally intelligent robot insofar as it detects and responds to people's feelings.

1999
- AIBO
  Sony launches first consumer robot pet dog AIBO (AI robot) with skills and personality that develop over time.

2002
- ROOMBA
  First mass produced autonomous robotic vacuum cleaner from iRobot learns to navigate and clean homes.

2011
- SIRI
  Apple integrates Siri, an intelligent virtual assistant with a voice interface, into the iPhone 4S.

2011
- WATSON
  IBM's question answering computer Watson wins first place on popular $1M prize television quiz show Jeopardy.

2014
- EUGENE
  Eugene Goostman, a chatbot passes the Turing Test with a third of judges believing Eugene is human.

2014
- ALEXA
  Amazon launches Alexa, an intelligent virtual assistant with a voice interface that completes shopping tasks.

2016
- TAY
  Microsoft's chatbot Tay goes rogue on social media making inflammatory and offensive racist comments.

2017
- ALPHAGO
  Google's A.I. AlphaGo beats world champion Ke Jie in the complex board game of Go, notable for its vast number ($2^{176}$) of possible positions.
What A.I. can do nowadays?

- Entertainment
- Writing
- Transport
- Healthcare
- Communications
- Arts & Cinema
**Artificial Intelligence vs Machine Learning vs Deep Learning**

- **Artificial Intelligence**: A technique which enables machines to mimic human behaviour.
- **Machine Learning**: Subset of AI technique which use statistical methods to enable machines to improve with experience.
- **Deep Learning**: Subset of ML which make the computation of multi-layer neural network feasible.
Searching Algorithms
Map Search vs Tree Search

Map Search vs Tree Search

Optimal!

Actions
- Suck
- Left
- Right
Exploring Trees
Exploring Trees

- **Uninformed search**
  - Breadth-first search
  - Depth-first search
  - Iterative deepening search

- **Informed search**
  - Greedy search
  - A* search

XKCD: Depth and Breadth

https://xkcd.com/2407/
Exploring Trees

- **Completeness:** The algorithm guarantees a solution if there is one.
- **Optimality:** The algorithm finds the solution that minimizes cost.
- **Time Complexity:** Time needed to find a solution.
- **Space Complexity:** Memory needed to perform the search.
Breadth-First Search

Queue

A

Explored
Breadth-First Search

Queue:
- B, C, D

Explored:
- A
Breadth-First Search

Queue
C, D, E, F

Explored
A, B
Breadth-First Search

Queue
D, E, F

Explored
A, B, C
Breadth-First Search

Queue
E, F, G, H, I

Explored
A, B, C, D
Breadth-First Search

Queue:
F, G, H, I, J, K

Explored:
A, B, C, D, E
Breadth-First Search

Queue
G, H, I, J, K

Explored
A, B, C, D, E, F
Breadth-First Search

Queue
H, I, J, K, L

Explored
A, B, C, D, E, F, G
Breadth-First Search

Queue
I, J, K, L, M, N

Explored
A, B, C, D, E, F, G, H
Queue
J, K, L, M, N

Explored
A, B, C, D, E, F, G, H, I
Breadth-First Search

Queue
K, L, M, N

Explored
A, B, C, D, E, F, G, H, I, J
Breadth-First Search

Queue
L, M, N

Explored
A, B, C, D, E, F, G, H, I, J, K
Breadth-First Search

Queue
M, N

Explored
A, B, C, D, E, F, G, H, I, J, K, L
Breadth-First Search

Queue

N

Explored

A, B, C, D, E, F, G, H, I, J, K, L, M
Breadth-First Search

Queue

Explored
A, B, C, D, E, F, G, H, I, J, K, L, M, N
Breadth-First Search

- Explores **shallow** nodes first.
- Is **complete** and **optimal**.
- Performance depending a lot on the **branching factor**.
- Takes very **long** to provide **deep solutions**.
- Potential **memory problem**.

<table>
<thead>
<tr>
<th>Depth</th>
<th>Nodes</th>
<th>Time</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>110</td>
<td>0.11 ms</td>
<td>107 KB</td>
</tr>
<tr>
<td>4</td>
<td>11,110</td>
<td>11 ms</td>
<td>10.6 MB</td>
</tr>
<tr>
<td>6</td>
<td>$10^6$</td>
<td>1.1 s</td>
<td>1 GB</td>
</tr>
<tr>
<td>8</td>
<td>$10^8$</td>
<td>2 min</td>
<td>103 GB</td>
</tr>
<tr>
<td>10</td>
<td>$10^{10}$</td>
<td>3 h</td>
<td>10 TB</td>
</tr>
<tr>
<td>12</td>
<td>$10^{12}$</td>
<td>13 days</td>
<td>1 PB</td>
</tr>
<tr>
<td>14</td>
<td>$10^{14}$</td>
<td>3.5 years</td>
<td>99 PB</td>
</tr>
<tr>
<td>16</td>
<td>$10^{16}$</td>
<td>350 years</td>
<td>10 EB</td>
</tr>
</tbody>
</table>

How scales a tree with branching factor 10...

Depth-First Search
Depth-First Search

Queue
B, C, D

Explored
A
Depth-First Search

Queue
E, F, C, D

Explored
A, B
Depth-First Search

Queue

J, K, F, C, D

Explored

A, B, E
Depth-First Search

Queue
K, F, C, D

Explored
A, B, E, J
Depth-First Search

Queue
- F, C, D

Explored
- A, B, E, J, K
Depth-First Search

Queue
C, D

Explored
A, B, E, J, K, F
Depth-First Search

Queue

D

Explored

A, B, E, J, K, F, C
Depth-First Search

Queue
G, H, I

Explored
A, B, E, J, K, F, C, D
Depth-First Search

Queue: L, H, I

Explored: A, B, E, J, K, F, C, D, G
Depth-First Search

Queue
H, I

Explored
A, B, E, J, K, F, C, D, G, L
Depth-First Search

Queue: M, N, I

Depth-First Search

Queue
N, I

Explored
Depth-First Search

Queue
I

Explored
Depth-First Search

Queue

Explored

Depth-First Search

- Explores **one branch** first.
- Is **not complete nor optimal**.
- Performance depending a lot on the **tree depth**.
- Can provide an **approximate solution** quite early on.
- Can **get lost** in **deeper parts** of the tree than the solution.
Iterative deepening Search

L = 0

Queue

Explored

A
Iterative deepening Search

Queue

Explored
A, B, C, D

L = 1
Iterative deepening Search

Queue

Explored

A, B, E, F, C, D, G, H, I

L = 2
Iterative deepening Search

Queue


Explored


L = 3
Iterative deepening Search

• **Combines** depth-first and breadth-first taking the **best of both worlds**.

• Is **complete** and **optimal**.

• Improves the **memory problem** of BFS.

• **Repeats** the **search** for shallow states (but the impact is not critical).
## Uninformed search comparative

<table>
<thead>
<tr>
<th>Search Method</th>
<th>Complete</th>
<th>Optimal</th>
<th>Time</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breadth-first</td>
<td>YES</td>
<td>YES</td>
<td>$O(b^d)$</td>
<td>$O(b^d)$</td>
</tr>
<tr>
<td>Depth-first</td>
<td>NO</td>
<td>NO</td>
<td>$O(b^m)$</td>
<td>$O(bm)$</td>
</tr>
<tr>
<td>Iterative deepening</td>
<td>YES</td>
<td>YES</td>
<td>$O(b^d)$</td>
<td>$O(bd)$</td>
</tr>
</tbody>
</table>

*b*: Branching factor.  
*d*: Depth where the solution is.  
*m*: Maximum depth of the tree.
Edge cost
Greedy Search
Greedy Search

Queue

B(3), C(2), D(2)

Explored

A
Greedy Search

Queue
B(3), D(2)

Explored
A, C
Greedy Search

Queue
B(3), G(6), H(7), I(10)

Explored
A, C, D
Greedy Search

Queue
G(6), H(7), I(10), E(5), F(12)

Explored
A, C, D, B
Greedy Search

Queue
G(6), H(7), I(10), F(12), J(8), K(9)

Explored
A, C, D, B, E
Greedy Search

Queue
H(7), I(10), F(12), J(8), K(9), L(13)

Explored
A, C, D, B, E, G
Greedy Search

Queue
I(10), F(12), J(8), K(9), L(13), M(14), N(9)

Explored
A, C, D, B, E, G, H
Greedy Search

Queue
I(10), F(12), K(9), L(13), M(14), N(9)

Explored
A, C, D, B, E, G, H, J
Greedy Search

Queue
I(10), F(12), L(13), M(14), N(9)

Explored
A, C, D, B, E, G, H, J, K
Greedy Search

Queue
I(10), F(12), L(13), M(14),

Explored
A, C, D, B, E, G, H, J, K, N
Greedy Search

Queue

F(12), L(13), M(14),

Explored

A, C, D, B, E, G, H, J, K, N, I
Greedy Search

Queue

Explored
A, C, D, B, E, G, H, J, K, N, I, F
L(13), M(14),
Greedy Search
Edge & Node cost

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>J</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

The table lists the values of each node.

The diagram represents the edge and node costs with values indicated on the edges and nodes.
### A* Search

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>M</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Queue**

- A(5)

**Explored**
A* Search

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>M</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Queue

B(7), C(5), D(4)

Explored

A
A* Search

<table>
<thead>
<tr>
<th>A = 5</th>
<th>B = 4</th>
<th>C = 3</th>
<th>D = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>E = 1</td>
<td>F = 0</td>
<td>G = 1</td>
<td>H = 2</td>
</tr>
<tr>
<td>I = 3</td>
<td>J = 4</td>
<td>K = 5</td>
<td>L = 6</td>
</tr>
<tr>
<td>M = 7</td>
<td>N = 8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Queue
- B(7), C(5), G(7), H(9), I(13)

Explored
- A, D
A* Search

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

Queue

B(7), G(7), H(9), I(13)

Explored

A, D, C
A* Search

<table>
<thead>
<tr>
<th>A = 5</th>
<th>B = 4</th>
<th>C = 3</th>
<th>D = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>E = 1</td>
<td>F = 0</td>
<td>G = 1</td>
<td>H = 2</td>
</tr>
<tr>
<td>I = 3</td>
<td>J = 4</td>
<td>K = 5</td>
<td>L = 6</td>
</tr>
<tr>
<td>M = 7</td>
<td>N = 8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Queue

G(7), H(9), I(13), E(6), F(12)

Explored

A, D, C, B
A* Search

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td></td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>J</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>K</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td></td>
<td>14</td>
<td></td>
</tr>
</tbody>
</table>

Queue
G(7), H(9), I(13), F(12), J(12), K(14)

Explored
A, D, C, B, E
A* Search

<table>
<thead>
<tr>
<th>A = 5</th>
<th>B = 4</th>
<th>C = 3</th>
<th>D = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>E = 1</td>
<td>F = 0</td>
<td>G = 1</td>
<td>H = 2</td>
</tr>
<tr>
<td>I = 3</td>
<td>J = 4</td>
<td>K = 5</td>
<td>L = 6</td>
</tr>
<tr>
<td>M = 7</td>
<td>N = 8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Queue
- H(9), I(13), F(12), J(12), K(14), L(19)

Explored
- A, D, C, B, E, G
A* Search

<table>
<thead>
<tr>
<th></th>
<th>A = 5</th>
<th>B = 4</th>
<th>C = 3</th>
<th>D = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>E = 1</td>
<td>F = 0</td>
<td>G = 1</td>
<td>H = 2</td>
<td></td>
</tr>
<tr>
<td>I = 3</td>
<td>J = 4</td>
<td>K = 5</td>
<td>L = 6</td>
<td></td>
</tr>
<tr>
<td>M = 7</td>
<td>N = 8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Queue
I(13), F(12), J(12), K(14), L(19), M(21), N(17)

Explored
A, D, C, B, E, G, H
### A* Search

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>M</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Queue**
- I(13), J(12), K(14), L(19), M(21), N(17)

**Explored**
- A, D, C, B, E, G, H, F
Guess the algorithm!
Guess the algorithm!
Guess the algorithm!
Guess the algorithm!
Travelling Salesman Problem (TSP)
"Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?"
TSP with 4 cities
TSP with 4 cities

- A
- B
- C
- D

Connections:
- A to B: 10
- B to C: 25
- C to D: 40
- D to A: 25
- A to C: 50
TSP with 4 cities

= 120  = 125  = 145  = 125  = 145  = 120
NP-Hard

Citi\es & Solutions & Array Sorting & Matrix Multiplication & NP Problems & P Problems & NP Complete & TSP & Graph Colouring
\hline
4 & 6 & & & & & & & \\
5 & 24 & & & & & & & \\
6 & 120 & & & & & & & \\
7 & 720 & & & & & & & \\
8 & 5040 & & & & & & & \\
9 & 40320 & & & & & & & \\
10 & 362880 & & & & & & & \\
20 & 121645100408832000 & & & & & & & \\
30 & $\sim 10^{30}$ & & & & & & & \\
40 & $\sim 10^{46}$ & & & & & & & \\
50 & $\sim 10^{62}$ & & & & & & & \\
100 & $\sim 10^{155}$ & & & & & & & \