Traveling Salesman Problem (TSP)

Abram Bradley and Karen Hughes

Test Questions

1. **What is the complexity of the brute force TSP and what two assumptions did we make?**
2. **What is a heuristic in the context of the TSP and which heuristic did you find most interesting?**
3. **For Ant Colony Optimization (ACO), what two factors does the probability of the next visited city depend on?**

Abram Bradley - Academic Background

- 2 year masters student of computer science: concentration software engineering
- Graduated from UT in 2016 with a B.A. in Neuroscience
- Currently a TA for cs102
- Formerly did research for Dr. Williams and Dr. Rhema Linder
- Formly did research for Dr. Penumadu in the Civil Engineering Department

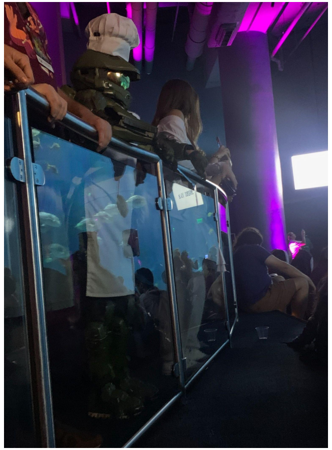
Favorite Languages and Technologies

- Favorite languages: TypeScript and Rust
- Favorite web frameworks React and Svelte

My food

- Broiled white fish
- Canadian bacon with fried eggs and cheddar on an everything bagel
- Tacos and roasted cauliflower
- Kabocha squash au gratin and Shepherd's pie
- Slow cooked elk chuck roast

- Atlanta
- dragon con
- Wet Leg
- Tern Club



New Orleans

- Frady’s Shrimp Po’ Boy
- Bywater Bakery
- Verti Marte Muffuletta
- Parleaux brewer and crawdad boil

- Favorite games
- Favorite shows

THE LEGEND OF ZELDA™ BREATH OF THE WILD

BARRY

RESIDENT EVIL™

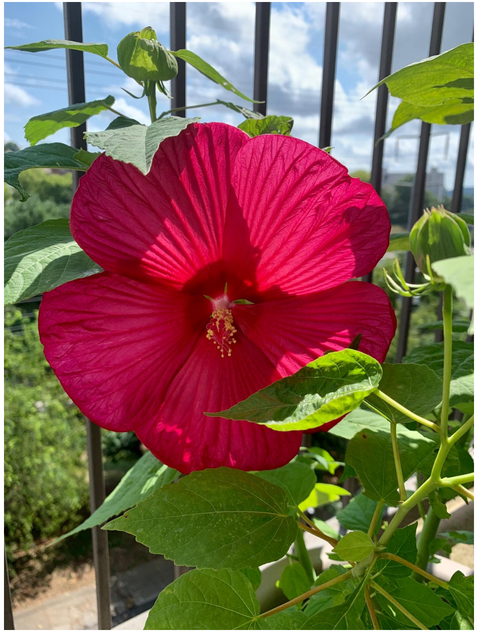
THE BOYS

DARK SOULS™

BETTER CALL Saul

Karen Hughes

- B.S./M.S. in Civil Engineering at UTK
- Pursuing M.S. in Computer Science
- GTA for COSC 102
- Software Engineering Intern at Trimble

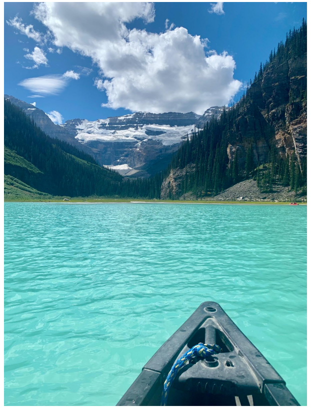


Trimble

Photo Credit https://www.trimble.com

Peyto Lake

Traveling, Kayaking, Hiking, Food, & Coffee



Lake Louise

Banff: Alberta, CA

Food Photo Credit: Google

Photography, Reading, & Board Games

Edwin Warner Park

Outline

- Overview
- History
- Algorithms
  - Brute Force
  - Dynamic Programming/Branch and Bound
  - Nearest Neighbor
  - Greedy
  - Nearest Insertion
  - Ant Colony Optimization
- Implementation
- Applications
- Open Issues
- Discussion

Overview: What is the Traveling Salesman Problem?

- Finding an optimal route through a set of points and returning to the start

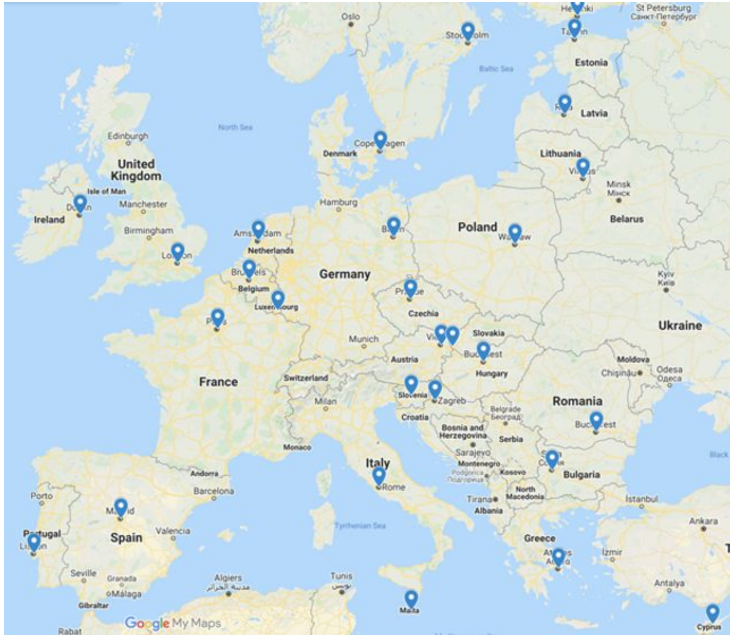
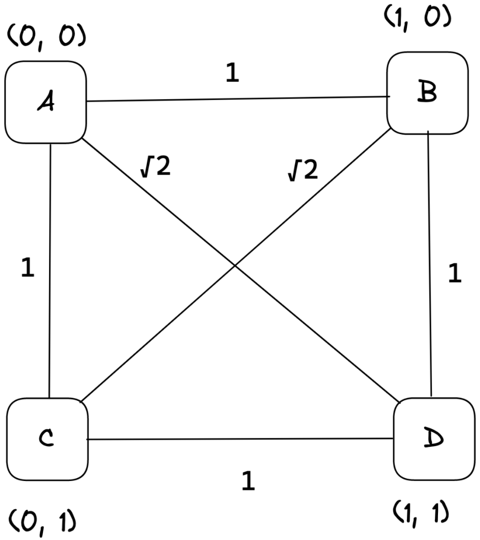


Photo Credit: https://www.oxbridgelaunchpad.com/post/ants-and-the-travelling-salesperson-problem

Overview: How to Represent this Problem as a Graph

- Represent each point in space as a 2-dimensional coordinate
  - A node in a **complete graph** with edges weighted by distance
- Use pythagorean theorem to calculate the distances between points
- Create an adjacency matrix
- Real world city problem:
  - Latitude and longitude for cities
  - Haversine Formula

$$d(c_1, c_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$



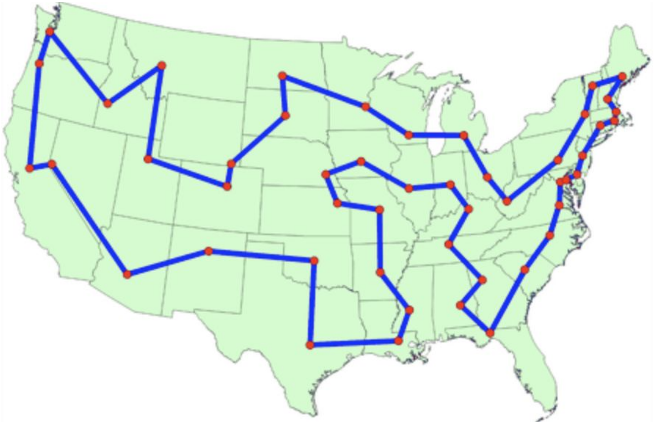
	A	B	C	D
A	0	1	1	√2
B	1	0	√2	1
C	1	√2	0	1
D	√2	1	1	0

Overview: Key Terms to Keep in Mind

- **Tour:** A closed path through a graph that starts and ends at the same point, vertices but not edges can be repeated
- **Hamiltonian Tour**: A closed path within a graph that visits every vertex exactly once and returns to the starting vertex also known as a Hamiltonian cycle
- **Symmetric**: Distance between any two cities or vertices are the same in both directions
- **Combinatorial Optimization**: branch of optimization focused on finding the best solution to a problem within a finite set of possible solutions
- **Heuristic**: A technique (or rule of thumb) designed for solving a problem quicker than classic methods, trading optimality for speed - e.g. a greedy approach
- **Superpolynomial:** Any big-O runtime increasing faster than n^k for any constant k

Overview: Why is TSP Important?

- Helps solve real world problems
  - Scheduling
  - Logistics



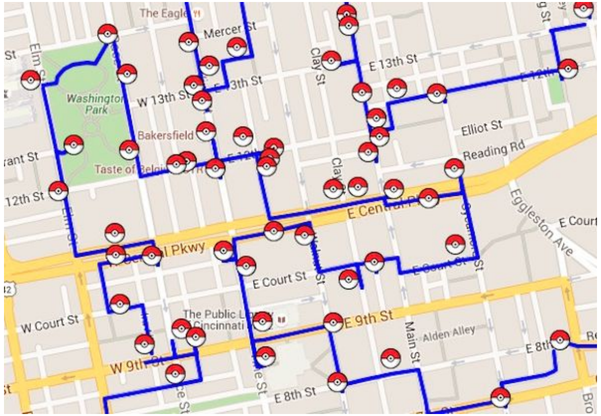


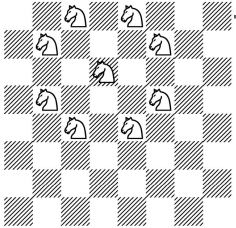
Photo Credit: https://www.math.uwaterloo.ca/tsp/poke/index.html ; https://optimization.cbe.cornell.edu/index.php?title=Traveling_salesman_problem

History: Traveling Salesman Problem


William Hamilton

- **1800s**:
  - William Hamilton and Thomas Kirkman make significant contributions the what would later become known as TSP
  - **Hamilton:** Hamiltonian cycle problem - finding a cycle that visits each vertex exactly once (Knight's Tour Puzzle)
  - **Kirkman:** combinatorial problem involved findi f permutations satisfying specific constraints


Thomas Kirkman



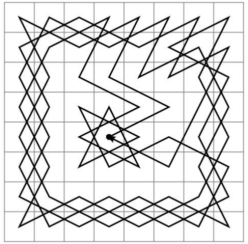


Photo Credit: https://www.sciencephoto.com/media/225986/view/william-hamilton-irish-mathematician ; https://mathshistory.st-andrews.ac.uk/Biographies/Kirkman/pictdisplay/ http://people.qc.cuny.edu/faculty/christopher.hanusa/courses/634sp13/Documents/634sp13ch2-4.pdf ; https://en.wikipedia.org/wiki/Knight%27s_tour

History: Traveling Salesman Problem

- **1930s**:
  - Karl Menger studies general form of TSP in mathematical terms
  - Hassler Whitney and Merrill Flood promote importance of the TSP
- **1950s**:
  - TSP gains traction among computer scientists and mathematicians in combinatorial optimization
  - The first major TSP solved was visiting 49 U.S. cities (G. Dantzig, R. Fulkerson, and S. Johnson, 1954)
- **1970s**:
  - Significant progress with algorithms to solve TSP
  - Development of Branch and Bound Algorithm

Algorithms: Techniques for Solving TSP

- Exact Algorithms - Finds the optimal solution to TSP
  - Typically computationally expensive and useful for smaller sets
- Heuristics - Finds a good solution to TSP
  - Does not guarantee optimal
  - Handles larger sets and can be faster than exact algorithms
  - Approximation ratio - helps find near optimal solutions to TSP by theoretically guaranteeing to be within a certain factor of optimal
    - Determined by analyzing performance
- Metaheuristics - a framework, rule set, or set of heuristics used together to find a good solution
  - E.G. simulating real world path-finding phenomenon such as ants foraging for food (more on this later)

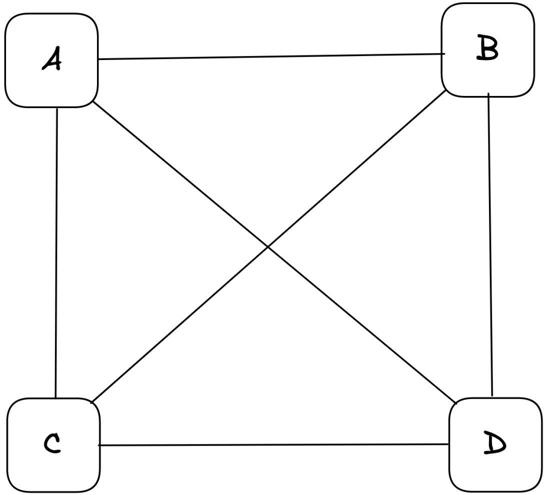
TSP Complexity

- Traveling salesman is NP-hard (not NP-complete)
- TSP is similar to the NP-complete decision algorithm of finding a Hamiltonian cycle (HAM)
- However TSP is an optimization problem of finding the **best** Hamiltonian cycle of a graph
- Criteria 1 for the TSP is the same as HAM, verify each node is visited once
- Criteria 2, verify there are no better solutions
- TSP could be made NP-complete by framing it differently such as, does a path exist < distance X, local optimal, but not global optimum

The Complexity of the Brute Force TSP

Take this k4 connect graph,

How many paths exist?



n! paths exist
4! = 24

```
[
  [ 'A', 'B', 'C', 'D' ], [ 'A', 'B', 'D', 'C' ],
  [ 'A', 'C', 'B', 'D' ], [ 'A', 'C', 'D', 'B' ],
  [ 'A', 'D', 'B', 'C' ], [ 'A', 'D', 'C', 'B' ],
  [ 'B', 'A', 'C', 'D' ], [ 'B', 'A', 'D', 'C' ],
  [ 'B', 'C', 'A', 'D' ], [ 'B', 'C', 'D', 'A' ],
  [ 'B', 'D', 'A', 'C' ], [ 'B', 'D', 'C', 'A' ],
  [ 'C', 'A', 'B', 'D' ], [ 'C', 'A', 'D', 'B' ],
  [ 'C', 'B', 'A', 'D' ], [ 'C', 'B', 'D', 'A' ],
  [ 'C', 'D', 'A', 'B' ], [ 'C', 'D', 'B', 'A' ],
  [ 'D', 'A', 'B', 'C' ], [ 'D', 'A', 'C', 'B' ],
  [ 'D', 'B', 'A', 'C' ], [ 'D', 'B', 'C', 'A' ],
  [ 'D', 'C', 'A', 'B' ], [ 'D', 'C', 'B', 'A' ]
]
```

The Complexity of the Brute Force TSP

However, we can make two assumptions about the TSP:

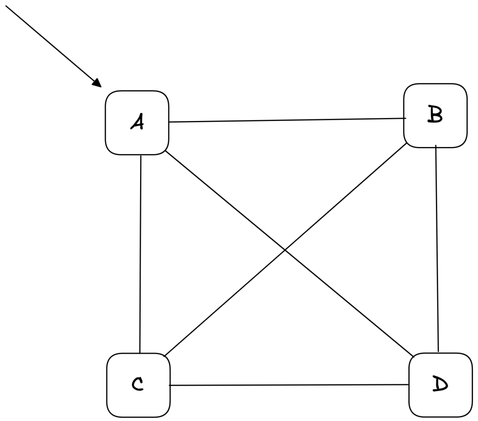
- The first assumption is common across all TSP variations
- The 2nd depends

The Complexity of the Brute Force TSP: Assumption 1

The start and end point are the same,

Hence n! Goes to (n-1)!

$(4 - 1)! = 6$




```
[ 'A', 'B', 'C', 'D', 'A' ]
[ 'A', 'B', 'D', 'C', 'A' ]
[ 'A', 'C', 'B', 'D', 'A' ]
[ 'A', 'C', 'D', 'B', 'A' ]
[ 'A', 'D', 'B', 'C', 'A' ]
[ 'A', 'D', 'C', 'B', 'A' ]
```

The Complexity of the Brute Force TSP: Assumption 2

- This is not true for all variations of the TSP, but is true for modeling real world routes.
- The distance from point A to B is the same as B to A
- e.g. `[ 'A', 'B', 'C', 'D', 'A' ] == [ 'A', 'D', 'C', 'B', 'A' ]`
- We can divide the number of tours by 2
- (4 - 1)! / 2 = 3

```
[ 'A', 'C', 'D', 'B', 'A' ]
[ 'A', 'D', 'B', 'C', 'A' ]
[ 'A', 'D', 'C', 'B', 'A' ]
```

- Brute force becomes infeasible beyond inputs of size 20
- (20 - 1) / 2 = 60,822,550,204,416,000
- The final complexity is (n - 1)! / 2

Complexity of Dynamic Programming & Branch and Bound

- Can we do better? Sort of, not really
- The runtime of these techniques are O(n^2 * 2^n)
- 20^2*(2^20) = 419,430,400
- Better than n!, but still NP-hard
- The takeaway, no good optimal solution algorithms to this problem exist

Heuristic Approaches

- Since finding an optimal solution is not feasible most of the time, what can we do?
- We can find “good enough” approximate solutions using heuristics
- Heuristics do not find the best solution, sometimes they do not even find a good solution, but they work well enough most of the time
- Nearest Neighbor
- Greedy Heuristic
- Ant Colony Optimization
- And many others!

Nearest Neighbors - (NN) Heuristic

- Time complexity O(n^2)
- This is probably the most straightforward approach
- Start at a point, go to the next nearest unvisited node, continue until returning to the start

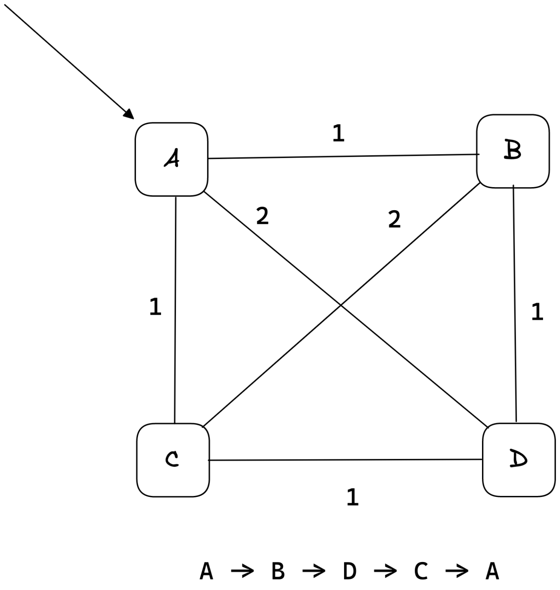
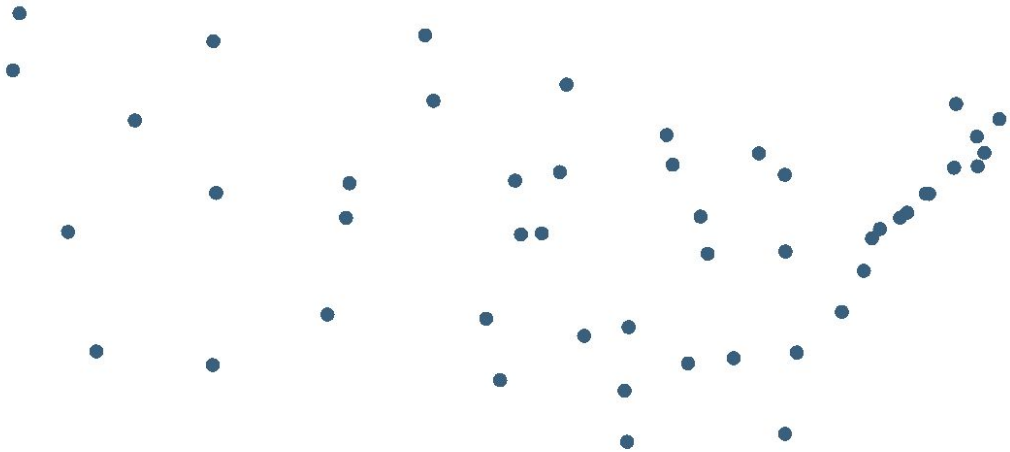


Photo credit: https://youtu.be/ncVgeRIcYsA, Chris Staecker, MA 15: Nearest neighbor algorithm

NN Example



(NN) Heuristic Caveats

- Depending on the dataset, starting at different points can produce dramatically different results
- Sometimes NN can even choose the worst tour
- A way we can improve this algorithm is running NN on each node
- This will increase the runtime to O(n^3), but will find the best solution using this technique

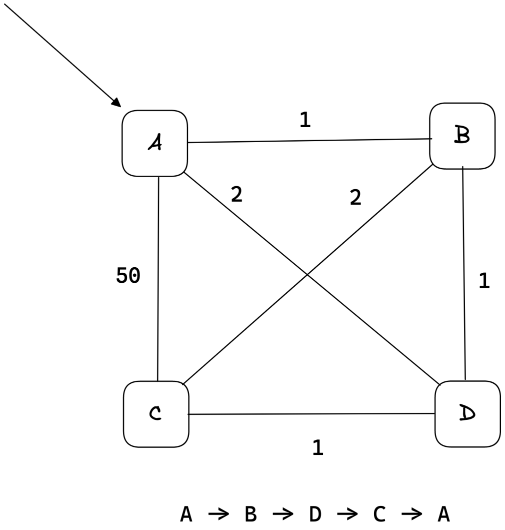
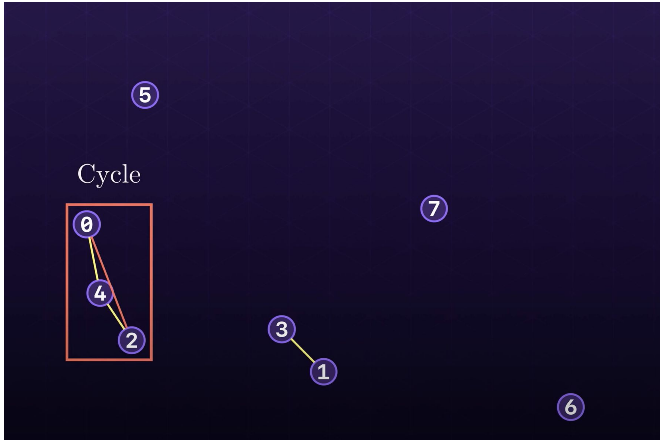


Photo credit: https://youtu.be/ncVgeRIcYsA, Chris Staecker, MA 15: Nearest neighbor algorithm

Greedy Heuristic

- Scan the whole graph and repeatedly add the edge of smallest cost
    - Reject for cycle < |V|, or edges that create vertices with degree > 2
- Similar technique to Kruskal's MST
- Slightly better than NN



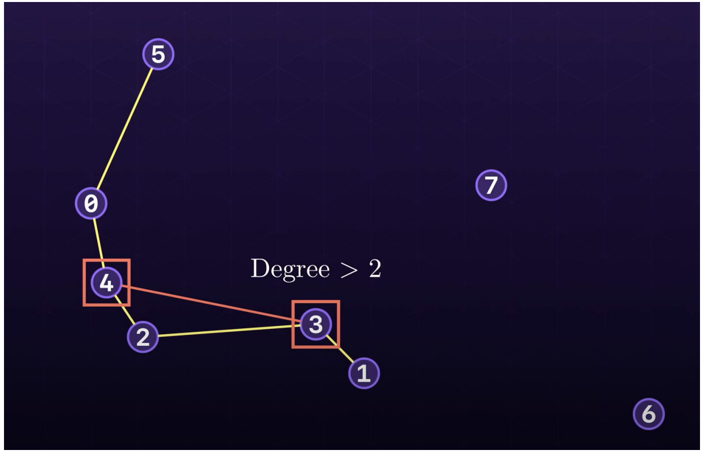
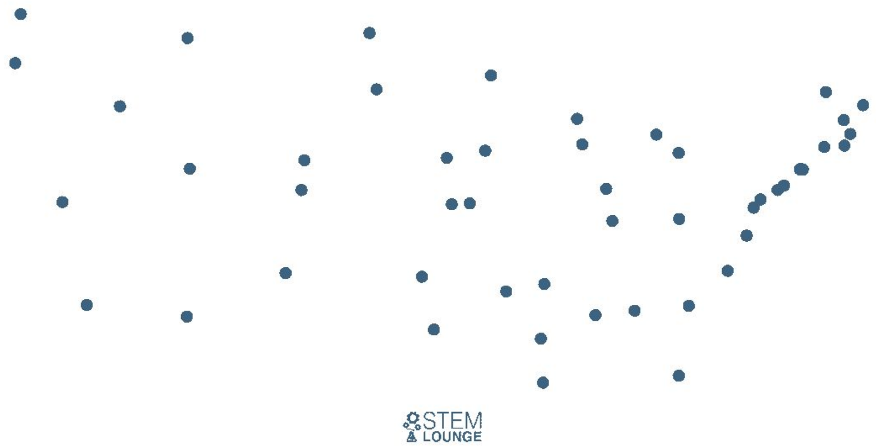


Photo credit: https://youtu.be/GiDsjIBOVoA, Redicible, The Traveling Salesman Problem: When Good Enough Beats Perfect

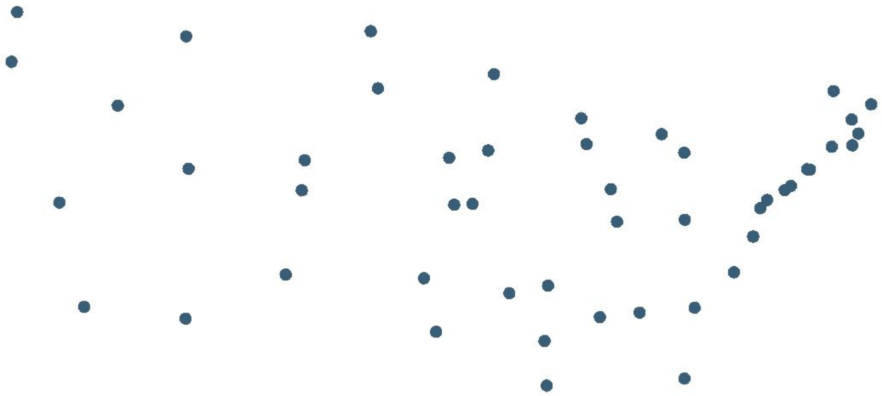
Greedy Heuristic Example



Nearest Insertion Heuristic

- Complexity n^3
- Starts by connecting two points
- Adds in the next point that will increase the cycle by the least
- Continues until all points are added
- Considerably more work per step, but tends to produce better results
- Same runtime as the repeated NN

Nearest Insertion Example



Techniques for Verifying Heuristic Solutions: Technique 1

- As stated before, TSP is NP-hard, meaning that if we have a best solution, the only way to verify it is brute forcing the actual solution
- We can brute force small inputs and run heuristics to check how well a heuristic performs

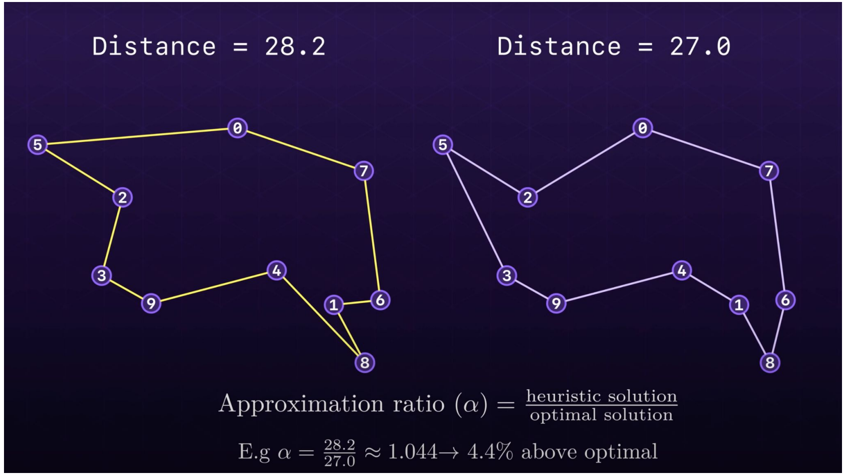


Photo credit: https://youtu.be/GiDsjIBOVoA, Redicible, The Traveling Salesman Problem: When Good Enough Beats Perfect

Verifying Heuristics Technique 2: MST Lower Bound

- How can heuristic solutions be verified on inputs too large to brute force?
- Those familiar with Prim's and Kruskal's MST will notice NN and greedy heuristic follow similar patterns
- There is a relationship between MST and TSP, turns out MST is a lower bound on the optimal TSP solution
- MST is a polynomial time algorithm and is easy to compute on large inputs
- We cannot know how far from the optimal an MST is, but it still gives something to check against

MST Lower Bound Example

- Intuition: if we take an optimal TSP tour and delete and edge, we are left with a spanning tree, but not an MST
- The MST is a lower bound on any spanning tree
- Optimal tour > spanning tree > MST

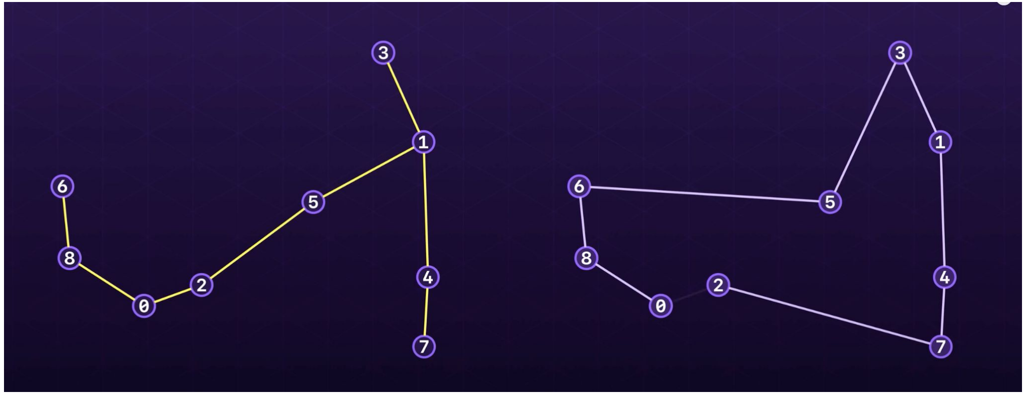
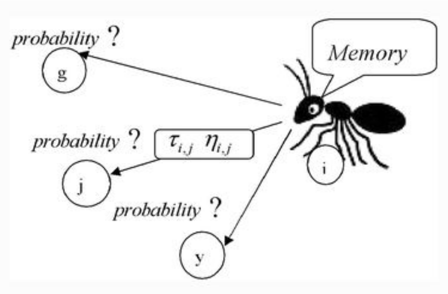


Photo credit: https://youtu.be/GiDsjIBOVoA, Redicible, The Traveling Salesman Problem: When Good Enough Beats Perfect

Ant Colony Optimization (ACO)

- ACO is inspired by ant behavior when foraging for food
  - Ants leave pheromones behind on their path
- Since these pheromones evaporate quickly, the highest concentration of pheromones make up the shortest route
- During each iteration, a tour is completed by each ant
- Each ant maintains already visited cities
- Probability of next city depends on:
  - Distance of current city to other cities
  - Intensity of local pheromone trail



$$p_{ij}^{k} = \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\prod_{l \in J_i^k}[\tau_{ij}(t)]^{\alpha}[\eta_{il}]^{\beta}}$$

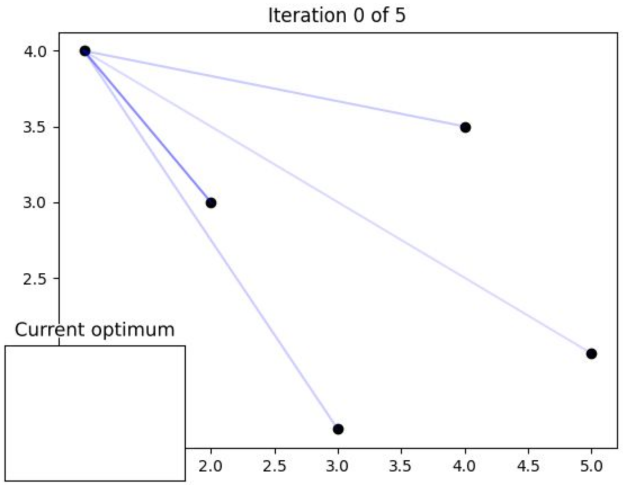
- Strength of pheromone trail from i to j
- Heuristics value for desirability and nearness
- Alpha and Beta - controls importance of each
- Sum of pheromone and heuristic values for all edges leaving node i

Photo Credit: https://www.washingtonpost.com/national/health-science/oh-to-be-an-ant-on-the- road/2015/05/22/eb6455ee-ff1c-11e4-8b6c-0dcce21e223d_story.html
https://springerplus.springeropen.com/articles/10.1186/s40064-016-2040-9

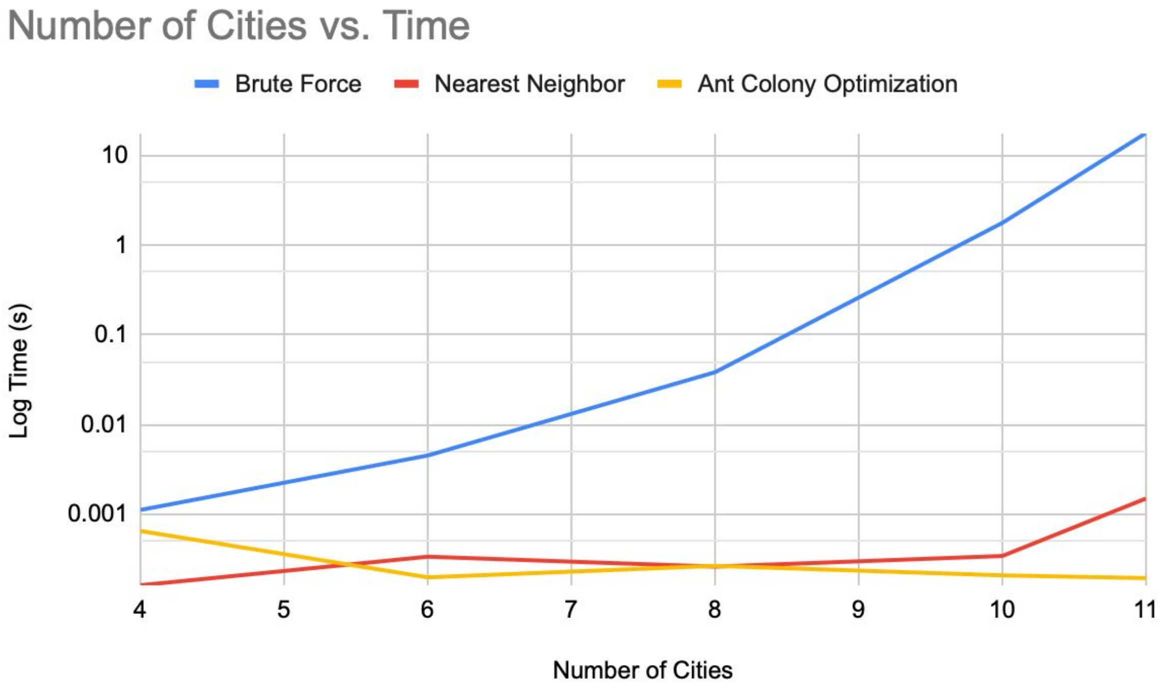
Ant Colony Optimization (ACO)

- Ants want to find the shortest path to food and make it back to their colony
- Behavior emerges out of all the individual ants working together
- Evaporation of pheromone trails diminish attractiveness of paths over time
- Finds the shortest path

  - ❑ Number of ants (m): explore search space to construct solutions to TSP instance
  - ❑ Each iteration (i) implements entire ACO algorithm with a starting pheromone trail
  - ❑ Repeated for a number of iterations and converges towards a good solution
  - ❑ Complexity: $O(i*n^2*m)$



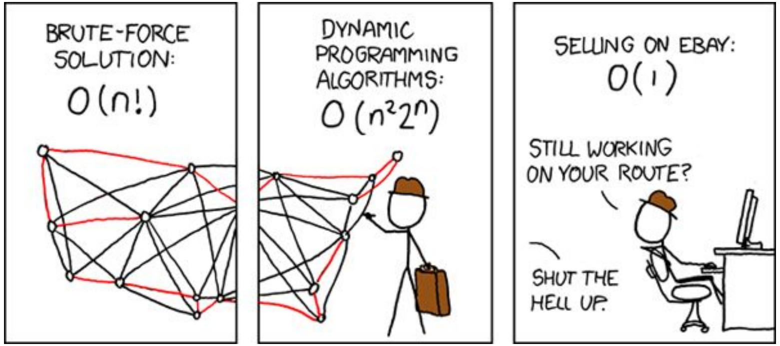
Implementations



Applications - Obvious Ones

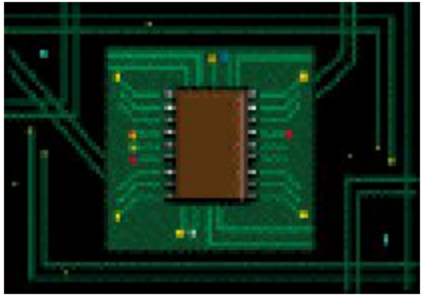
Transportation:

- Vehicle routing
- School bus routing
- Public transportation
- Mail delivery: USPS, FedEx, UPS
- Ride sharing services: Uber and Lyft
- Navigation and mapping companies: Google maps



Applications

- Concorde's implementation to optimize scan chains in integrated circuits
  - Scan chains (scan paths) is a design technique used to test and debug of integrated circuits



- Fiber optical networks design at Bell Communications Research
  - Routing of sonet rings, which provide communications links through a set of sites

Photo Credit: https://www.math.uwaterloo.ca/tsp/index.html

Open Issues

- Exact Algorithms: limited to small and medium scale problems
  - Continued work in finding an practical exact algorithm to solve large-scale problems
- Approximation ratio limitations
  - Approximation ratios are at times not practical for specific problems
  - Continued work in developing more accurate and faster algorithms
- Finding efficient algorithms to solve real world TSP problems
  - Real world problems may have the cost of tour change over time
  - Real world problems may have multiple objectives
- Concorde TSP Solver: 85,900-city, largest TSPLIB challenge problem
  - Developed early 1990s, widely regarded as one of the most powerful TSP solver

References

- https://books.google.com/books?hl=en&lr=&id=gKWdDwAAQBAJ&oi=fnd&pg=PA1&dq=traveling+salesman+problem&ots=a9gvV15lJ9&sig=dAHrh3SoCd-9Da0nsDXGL8-nM2Q#v=onepage&q=traveling%20salesman%20problem&f=false
- http://seor.vse.gmu.edu/~khoffman/TSP_Hoffman_Padberg_Rinaldi.pdf
- https://www.math.uwaterloo.ca/tsp/index.html
- https://www.sciencedirect.com/science/article/pii/S0167637703000932?casa_token=FbCyw4aoJvQAAAAA:CvWW1Q2LjCFyJCUiW80ZCbDVaF4JvxP69iHSI3IUw3N-soHKLR7FaGLekt4RJ-QHP4Ytq8q3
- https://www.oxbridgelaunchpad.com/post/ants-and-the-travelling-salesperson-problem
- https://www.sciencedirect.com/science/article/pii/S0167637708001132?casa_token=pUxiLBym3AgAAAAA:25C9mtOd28A4Enb5XRM9_WPRkSv-ka2-2w9cbLLorwmWKUi8di0NkiQxAmCwp18aHUVUITT0
- https://wahidulalamriyad.medium.com/optimization-algorithms-for-compiling-tsp-b86aea307a16
- https://www.sciencedirect.com/science/article/pii/S1572528608000339
- https://www.researchgate.net/profile/Zinaid-Kapic/publication/346041240_Implementation_of_the_genetic_algorithm_in_an_application_for_solving_a_traveling_salesman_problem/links/5fb8166da6fdcc6cc6540d9e/Implementation-of-the-genetic-algorithm-in-an-application-for-solving-a-traveling-salesman-problem.pdf
- https://youtu.be/GiDsjIBOVoA, Redicible, The Traveling Salesman Problem: When Good Enough Beats Perfect
- https://youtu.be/ncVgeRIcYsA, Chris Staecker, MA 15: Nearest neighbor algorithm
- https://www.youtube.com/watch?v=2jncD54ryGs, Eddie Woo: The Travelling Salesman (3 of 3: Ant Colonisation Heuristic)
- Collective Systems, Biologically Inspired Computation, Dr. Schuman
- https://stemlounge.com/animated-algorithms-for-the-traveling-salesman-problem/
- https://pypi.org/project/sko/

Discussion

- Questions?

Test Questions Revisited

1. **What is the complexity of the brute force TSP and what two assumptions did we make?**
2. **What is a heuristic in the context of the TSP and which heuristic did you find most interesting?**
3. **For Ant Colony Optimization (ACO), what two factors does the probability of the next visited city depend on?**