Robert Sedgewick | Kevin Wayne

http://princeton.edu/~cos126

Assignment 8 Tips and Tricks

- *traveling salesperson problem*
- *nearest insertion heuristic*
- *smallest increase heuristic*
- *implementation*
- *beyond*

Kevin Wayne
Last updated on 4/20/17 1:25 PM

Goals

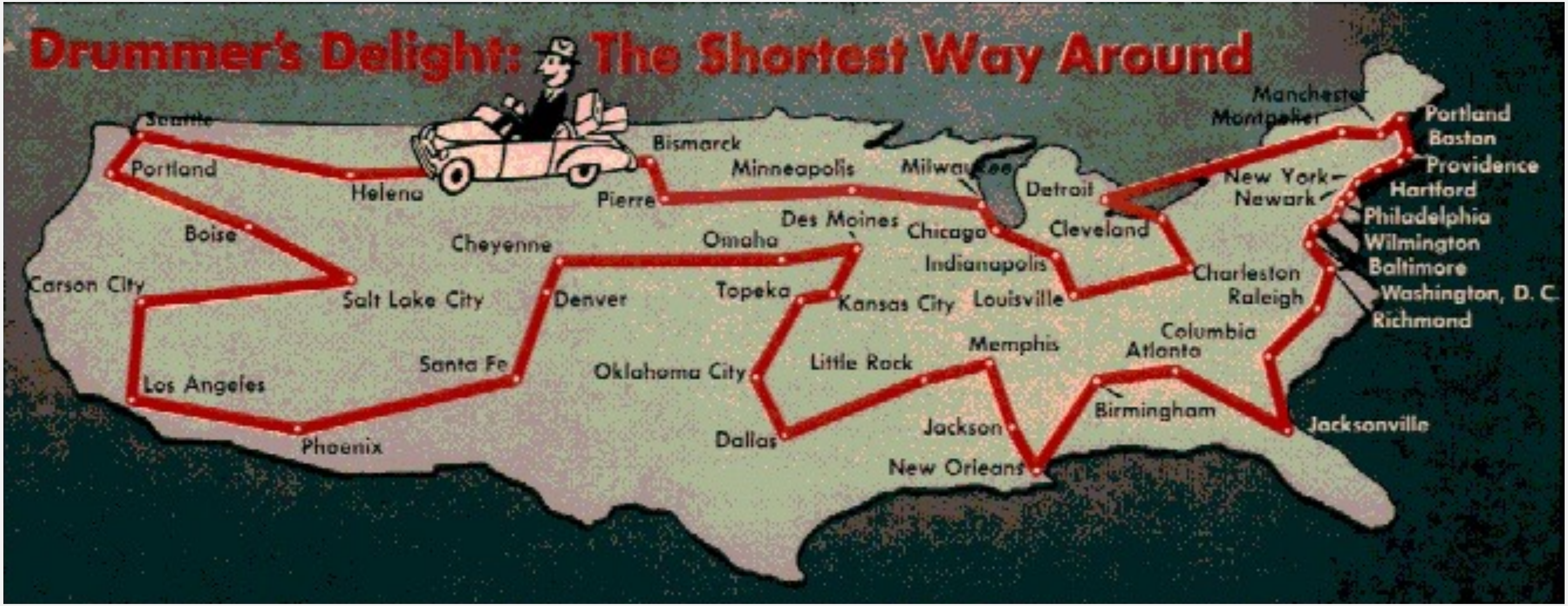
- Implement a data structure using linked lists.
- Analyze the running time of a program.
- Explore a notoriously difficult problem (see Intractability lecture).

Assignment 8 Tips and Tricks

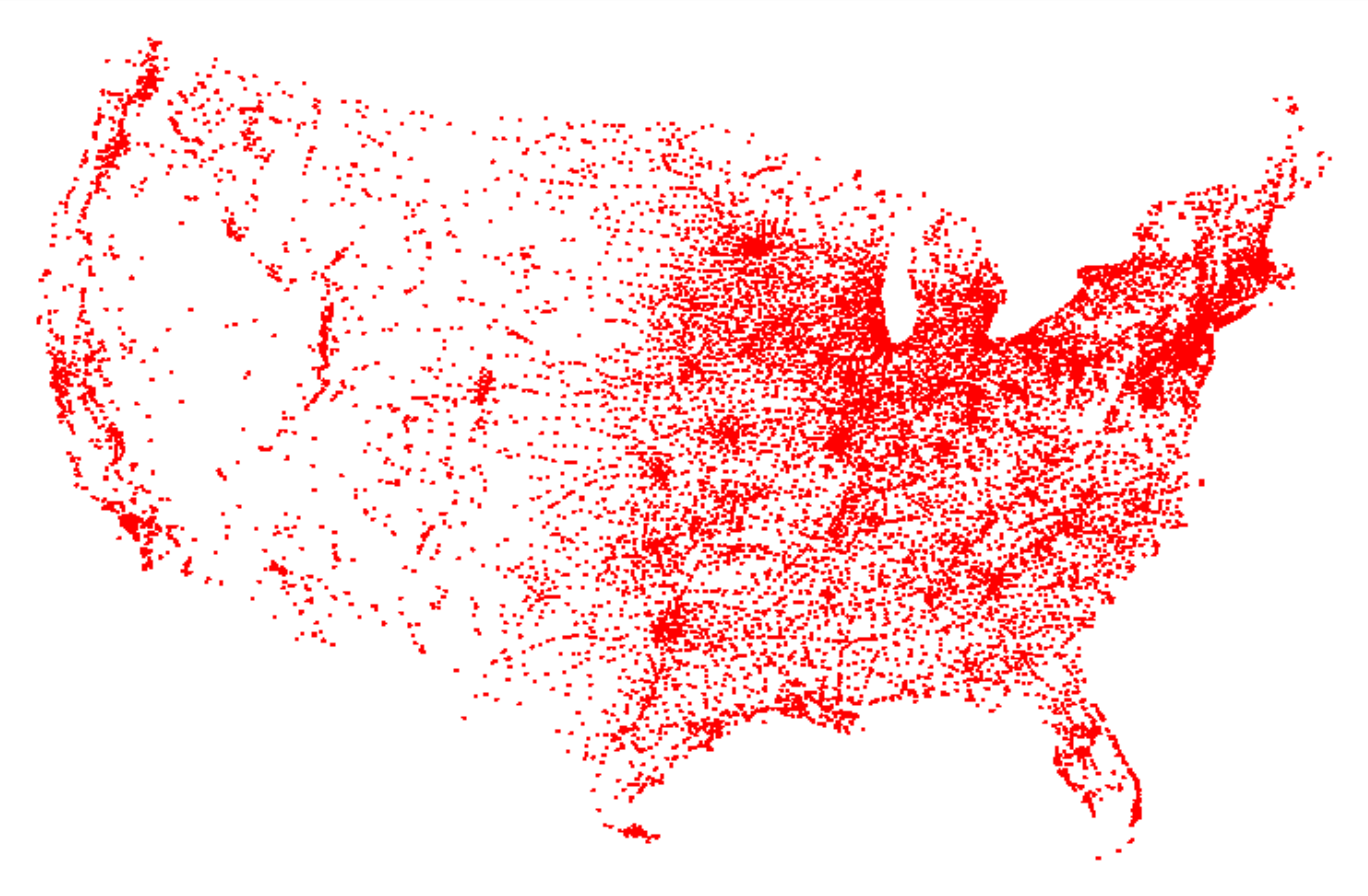
- *traveling salesperson problem*
- *nearest insertion heuristic*
- *smallest increase heuristic*
- *implementation*
- *beyond*

Euclidean TSP

Given n points in the plane, find a tour of minimum length that visits them all.



USA cities



13,509 cities in the United States
http://www.tsp.gatech.edu

USA cities

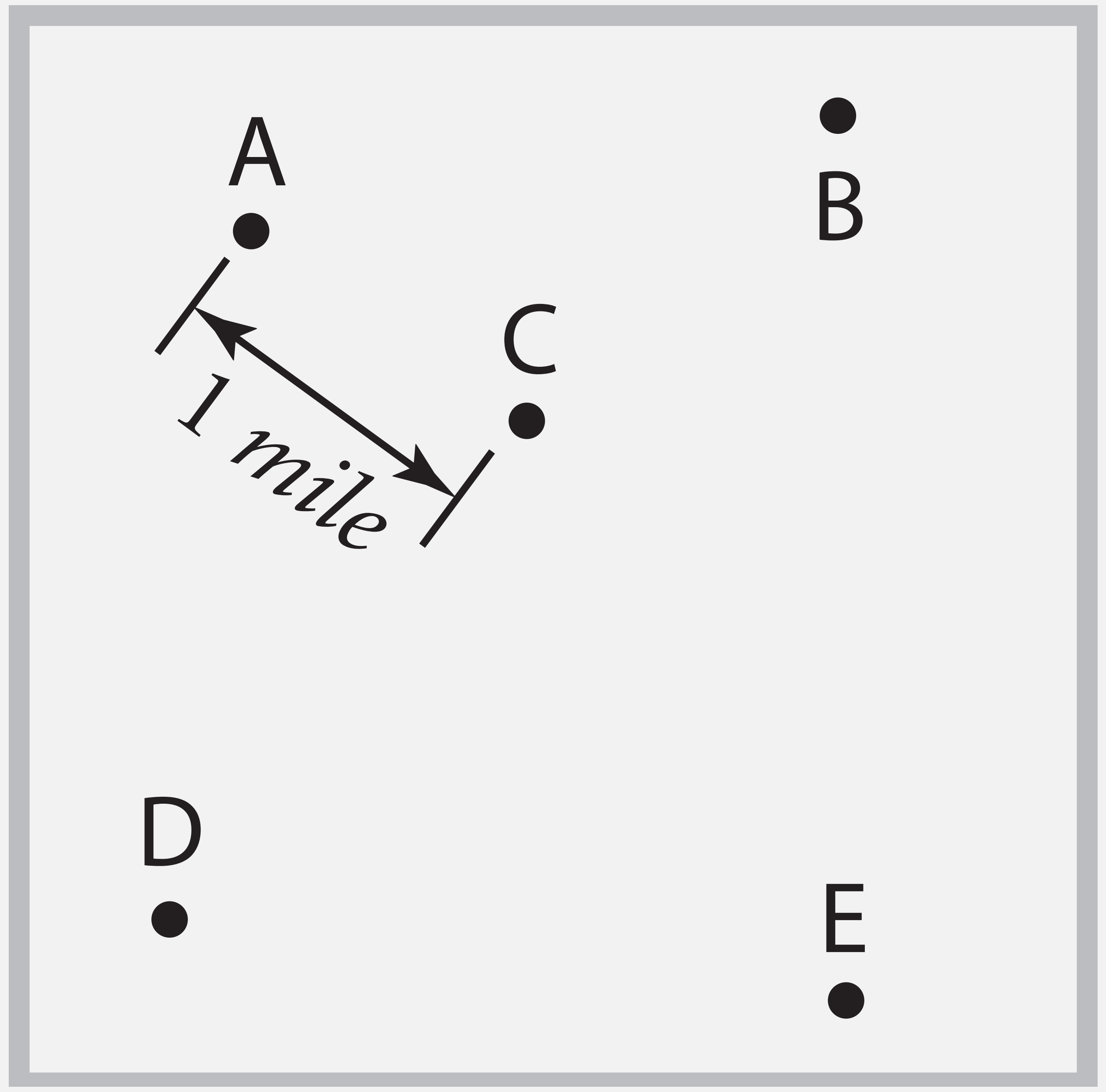
optimal TSP tour
http://www.tsp.gatech.edu

General TSP

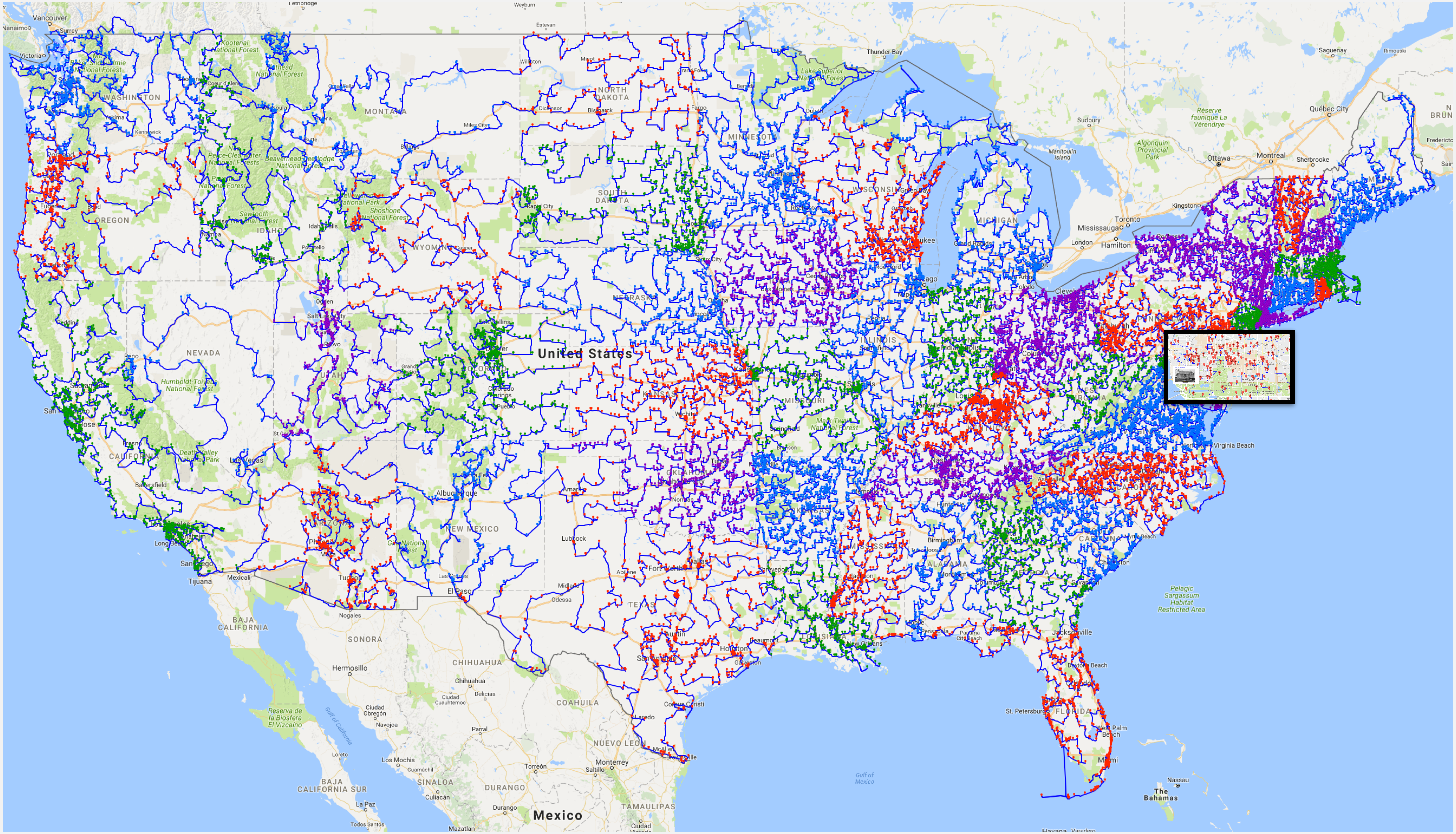
Given n points and pairwise "distances", find a tour of minimum length that visits them all.

Distances could represent:

- Costs.
- Travel times.
- Fuel consumed.
- Some more abstract quantity.
- ...

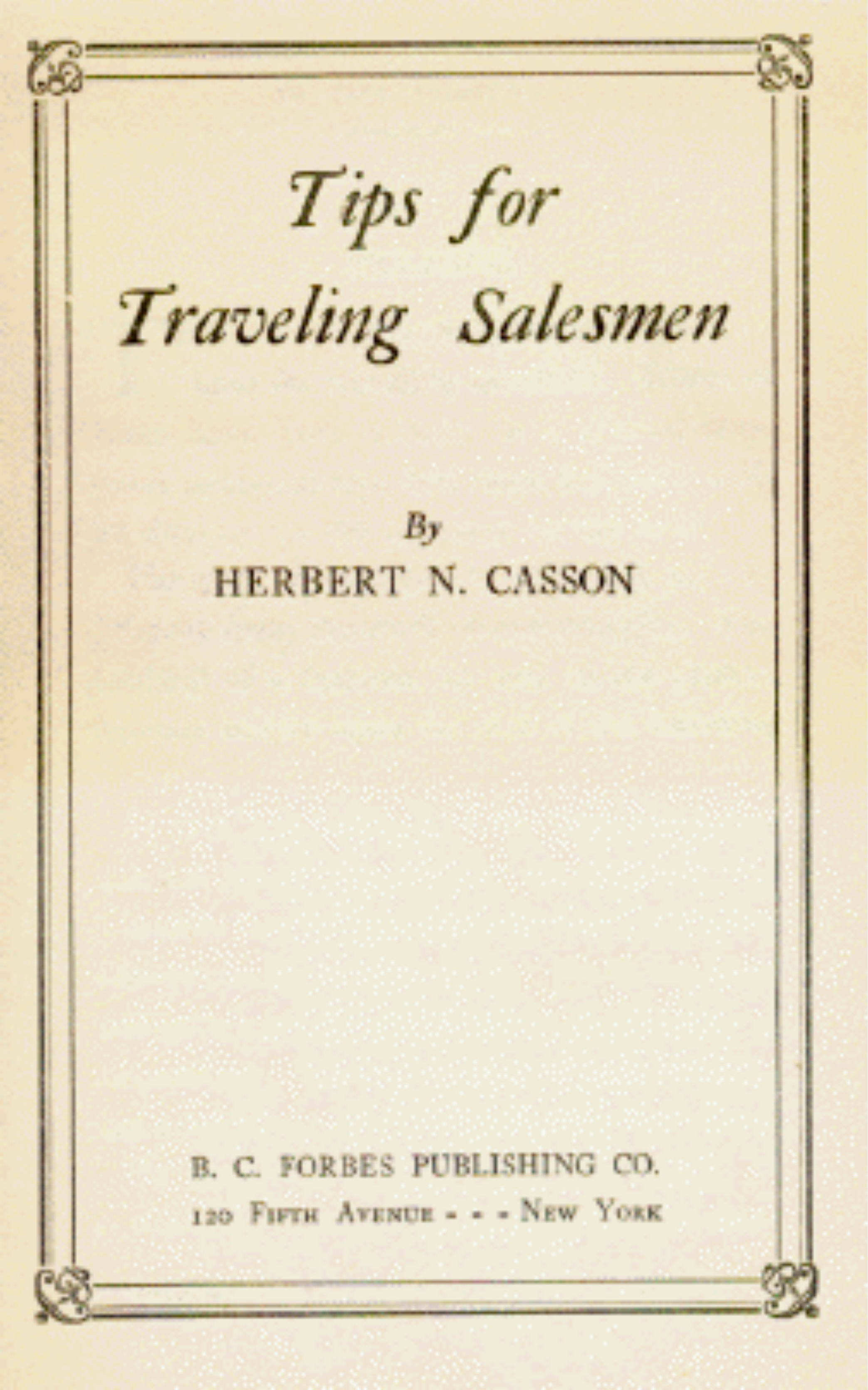


USA landmarks



Applications

Traveling salespeople? Probably not.

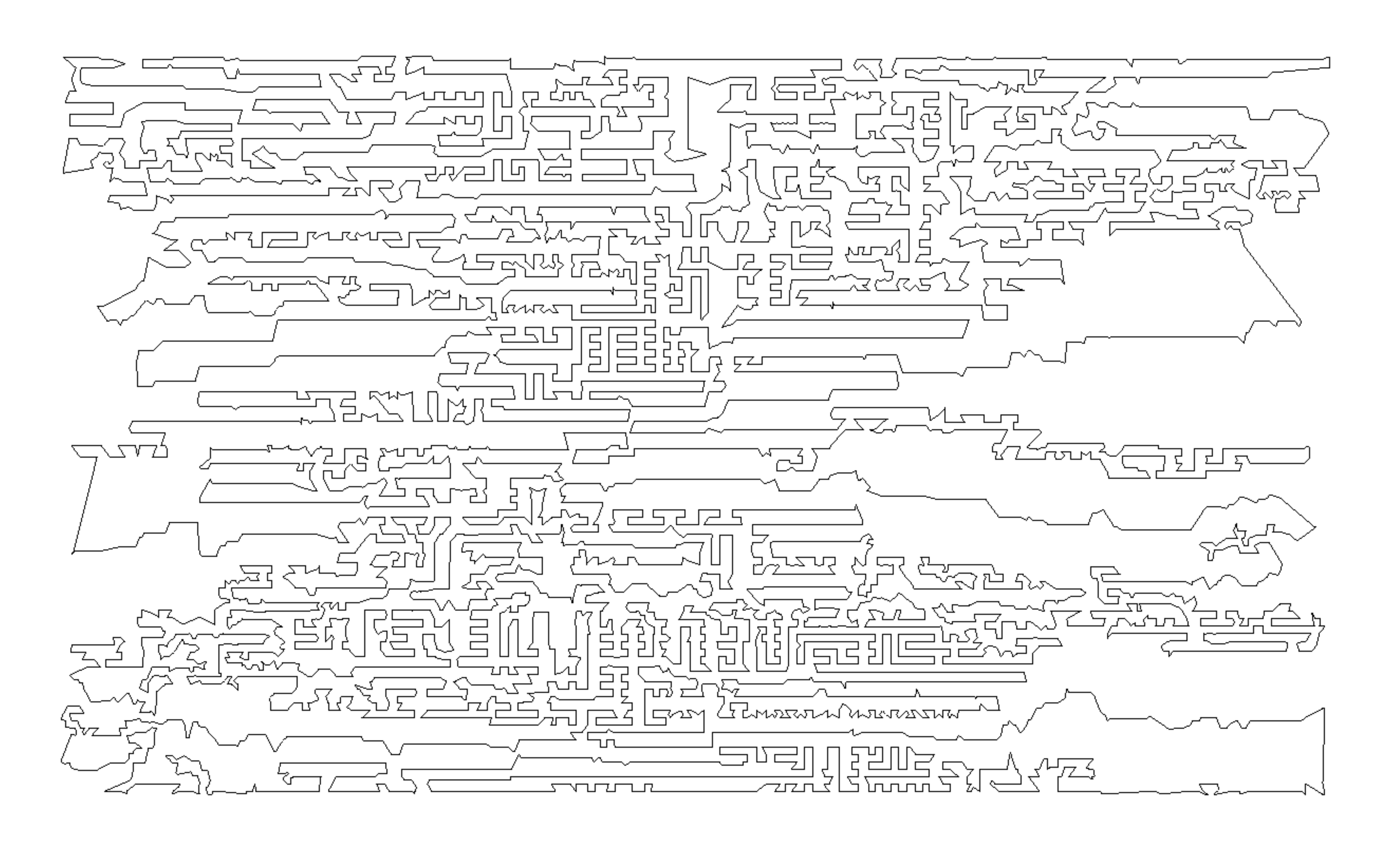


Applications

VLSI design. Drill holes in a printed circuit board.

Applications

VLSI design. Drill holes in a printed circuit board.



Applications

VLSI design. Drill holes in a printed circuit board.

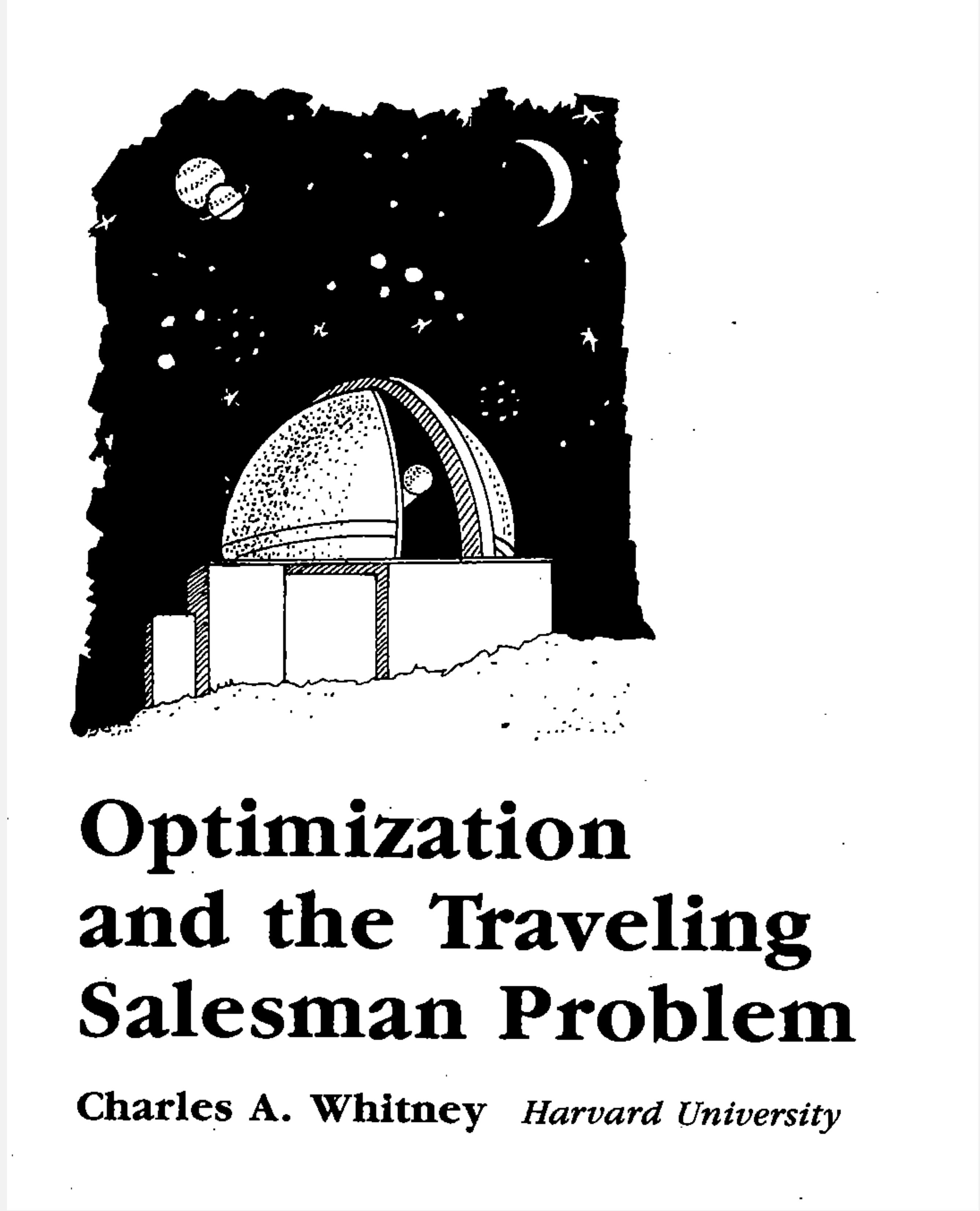
Space telescope. Re-position satellite to image celestial objects.

Fuel-Saving Strategies for Dual Spacecraft Interferometry Missions

Christopher A. Bailey,[1] Timothy W. McLain,[2] and Randal W. Beard[3]

Abstract

Separated spacecraft interferometry missions will require that spacecraft move in a coordinated fashion to ensure minimal and balanced consumption of fuel. This paper develops strategies for determining interferometry mission plans that result in significant fuel savings over standard approaches. Simulation results demonstrate that valuable reductions in fuel consumption can be realized by combining the retargeting and imaging maneuvers required to image multiple stellar sources. Fuel-optimal imaging strategies have been developed for two-spacecraft interferometry missions similar to the proposed StarLight mission using chained local optimization methods. Based on these strategies, sampling-pattern guidelines for space-borne interferometry missions have been developed.



Applications

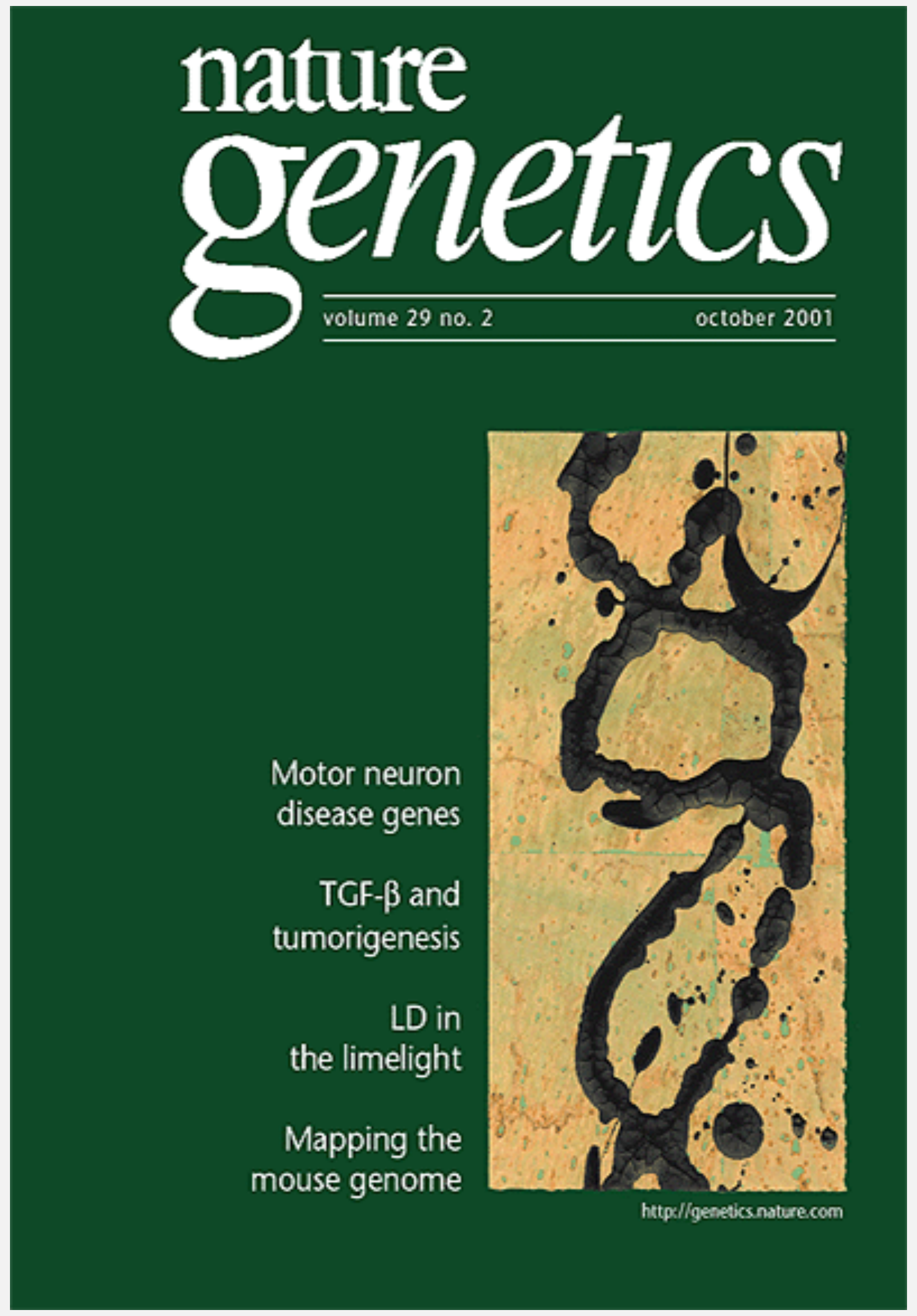
VLSI design. Drill holes in a printed circuit board.

Space telescope. Re-position satellite to image celestial objects.

Computational biology. Map the mouse genome.

A radiation hybrid transcript map of the mouse genome

Philip Avner[1,2], Thomas Bruls[1], Isabelle Poras[1], Lorraine Eley[3], Shahinaz Gas[1], Patricia Ruiz[4], Michael V. Wiles[5,10], Rita Sousa-Nunes[6], Ross Kettleborough[6], Amer Rana[6], Jean Morissette[4], Liz Bentley[3], Michelle Goldsworthy[3], Alison Haynes[3], Eifion Herbert[3], Lorraine Southam[3], Hans Lehrach[5], Jean Weissenbach[1], Giacomo Manenti[7], Patricia Rodriguez-Tome[8,10], Rosa Beddington*, Sally Dunwoodie[6,9] & Roger D. Cox[3]



Applications

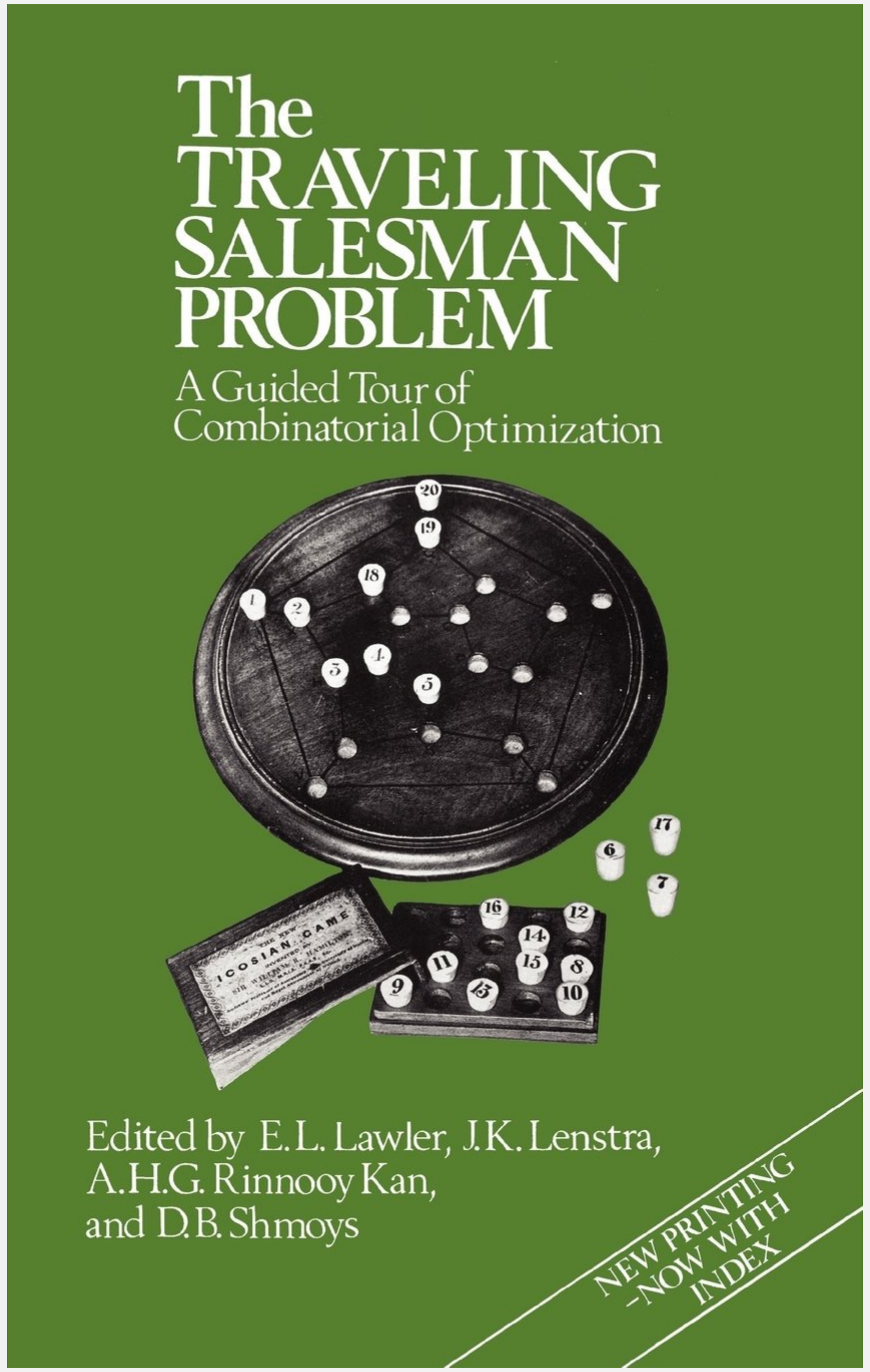
VLSI design. Drill holes in a printed circuit board.

Space telescope. Re-position satellite to image celestial objects.

Computational biology. Map the mouse genome.

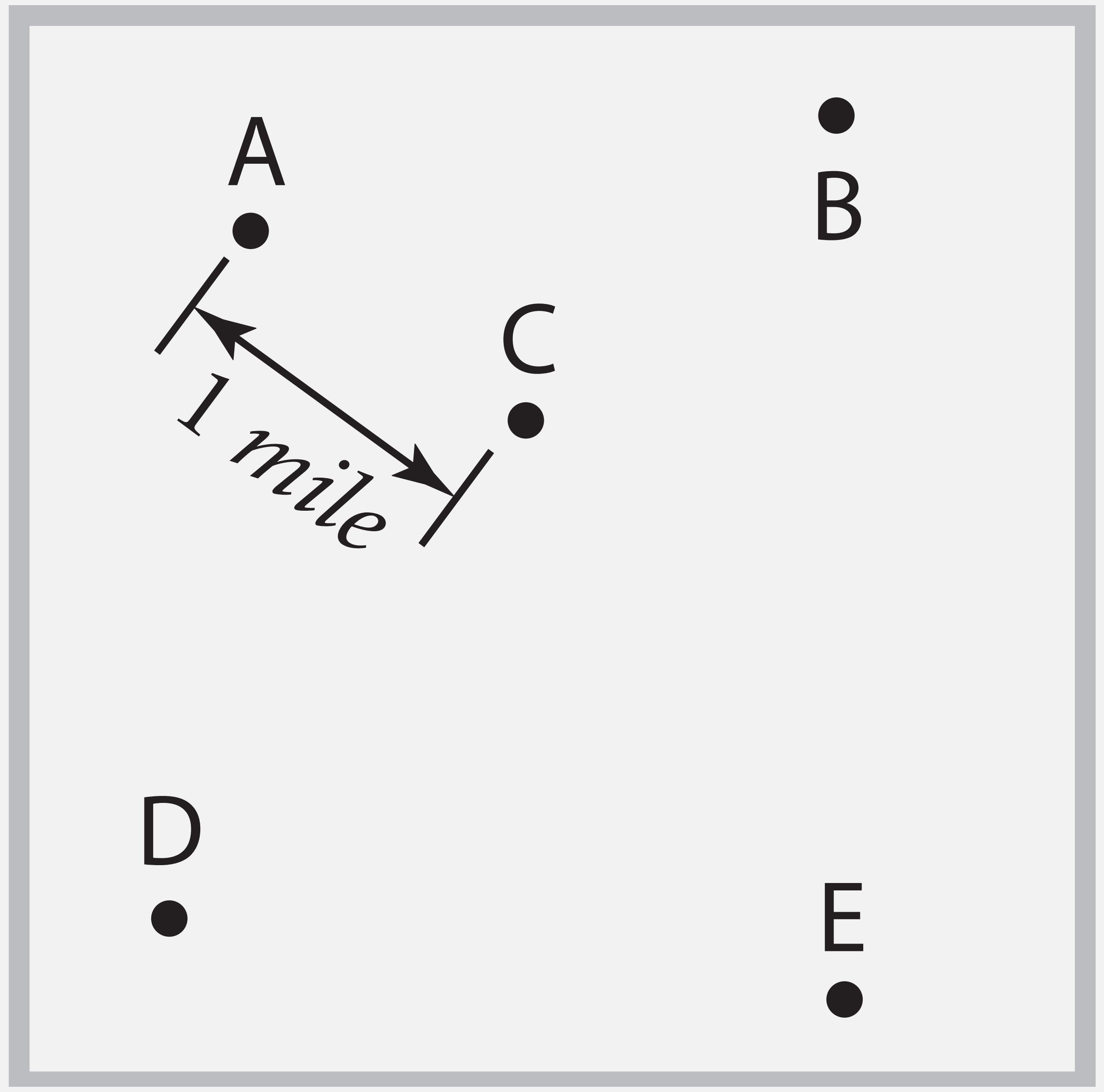
Combinatorial optimization. Training ground for new techniques.

seminal problem
in operations research

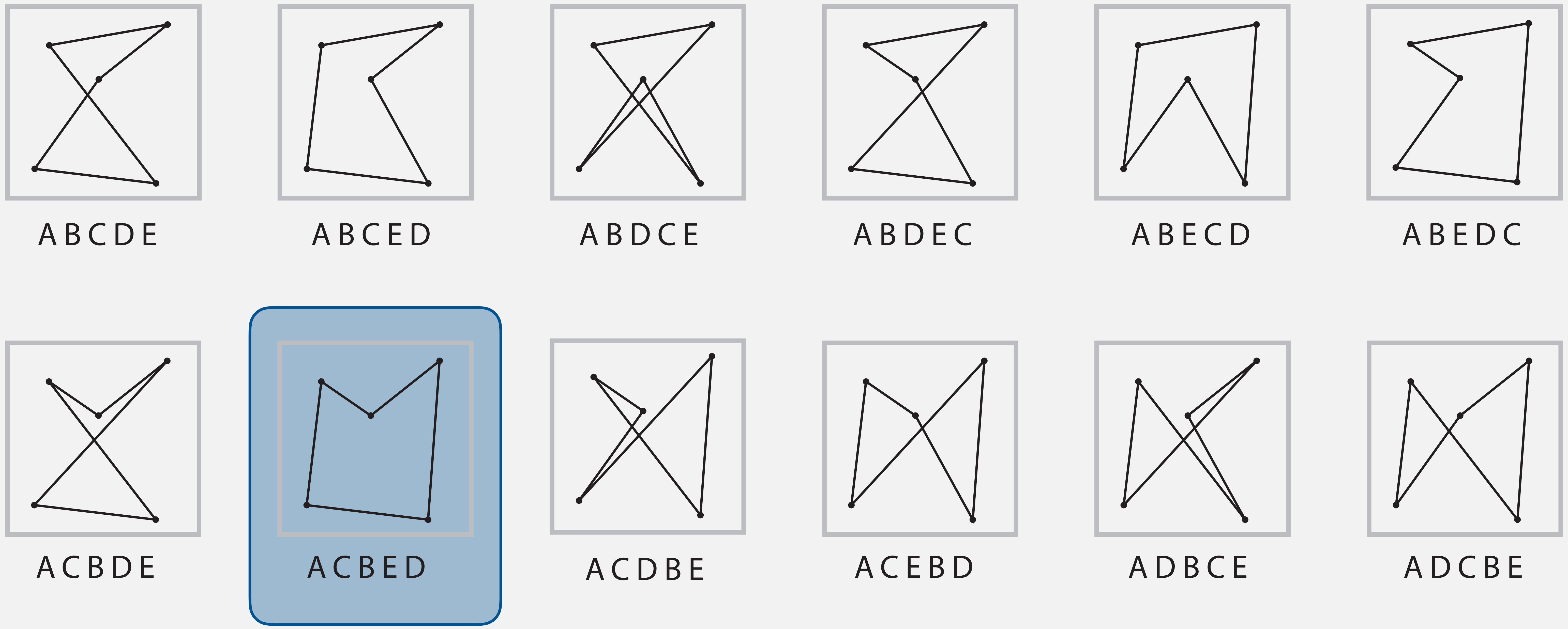


Brute-force algorithm

Easy? Try all possible tours; pick best one.



all possibilities



Brute-force algorithm

Estimate how many possible tours among $n = 1{,}000$ cities?

A. 181,440

B. 1,000,000

C. 60,822,550,204,416,000

D. $2.01 \times 10^{2{,}564}$

QuizSocket.com

JYZLR3 Join Quiz

n	# tours
5	12
10	181,440
20	60,822,550,204,416,000
50	3.04×10^{62}
100	4.66×10^{155}
1,000	$2.01 \times 10^{2{,}564}$

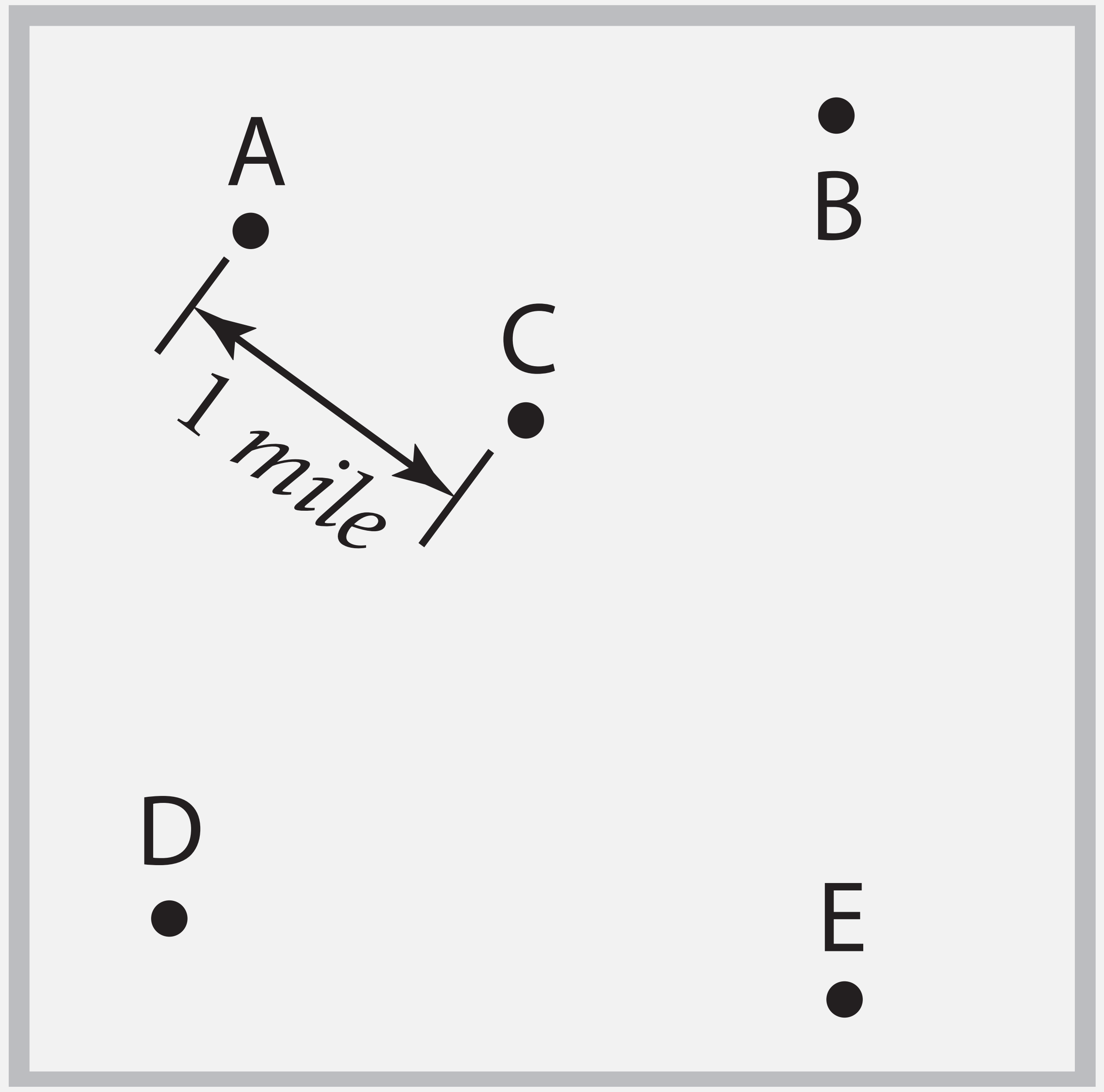
Brute-force algorithm

Easy? Try all possible tours; pick best one.

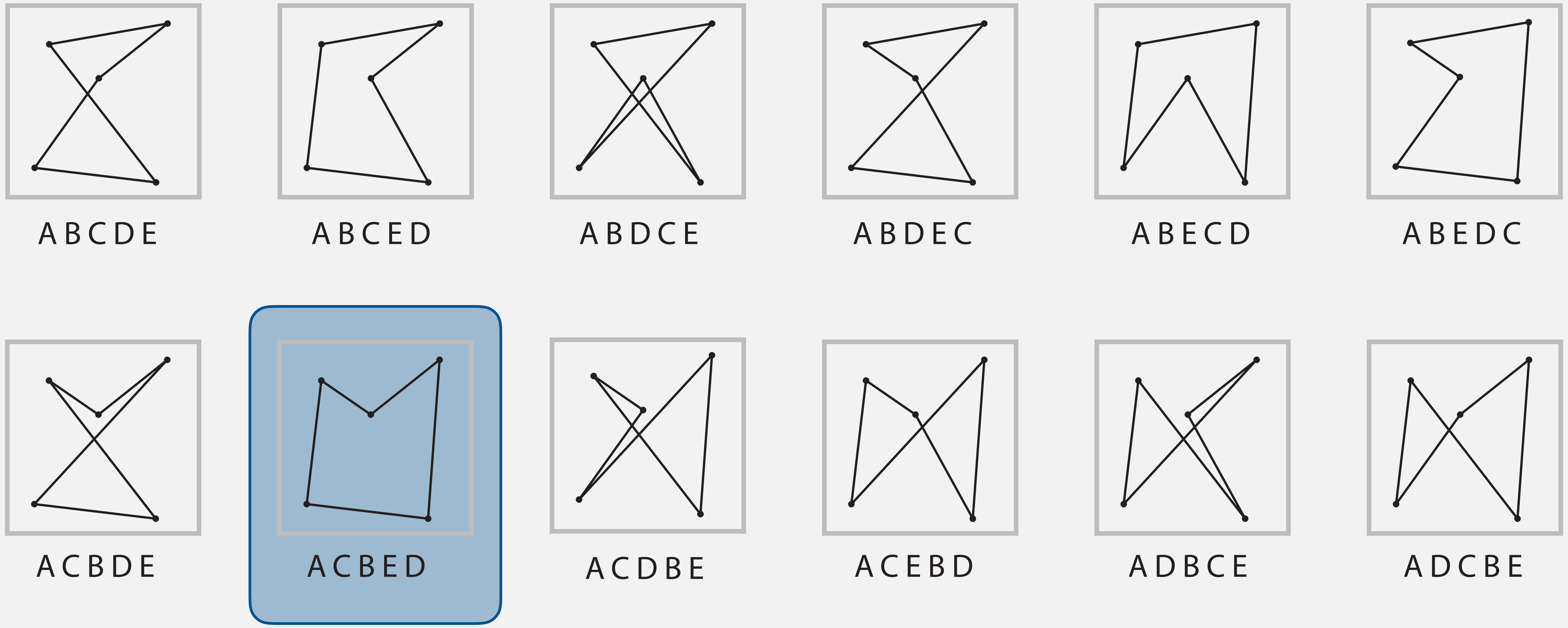
ABCDE, *BCDEA*, *CDEAB*, *DEABC*, and *EABCD* are the same tours

Hard? Given n points, number of possible tours is $\dfrac{n \times (n-1) \times (n-2) \times \cdots \times 1}{2}$

ABCDE and *AEDCB* are the same tours



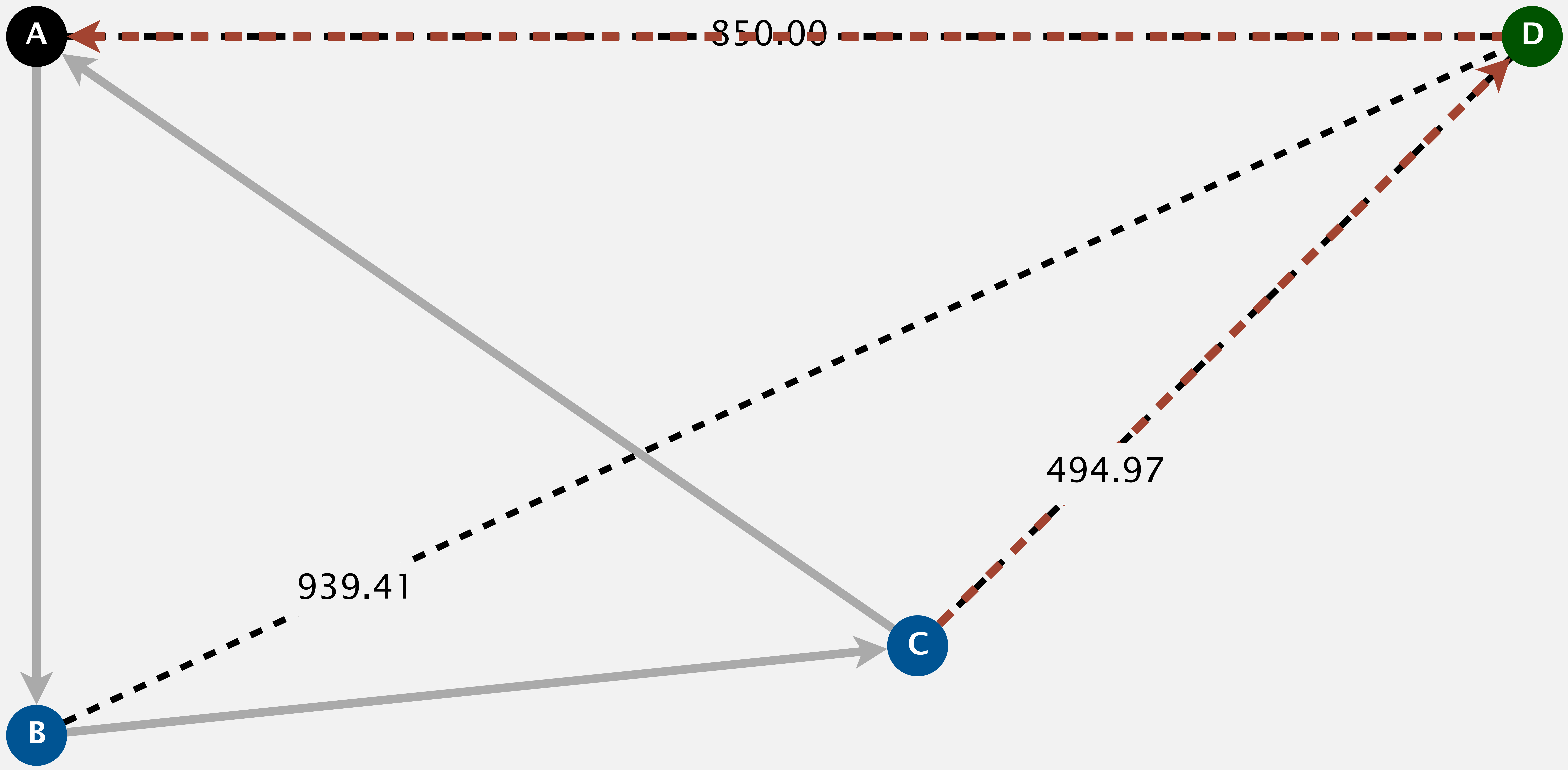
all possibilities



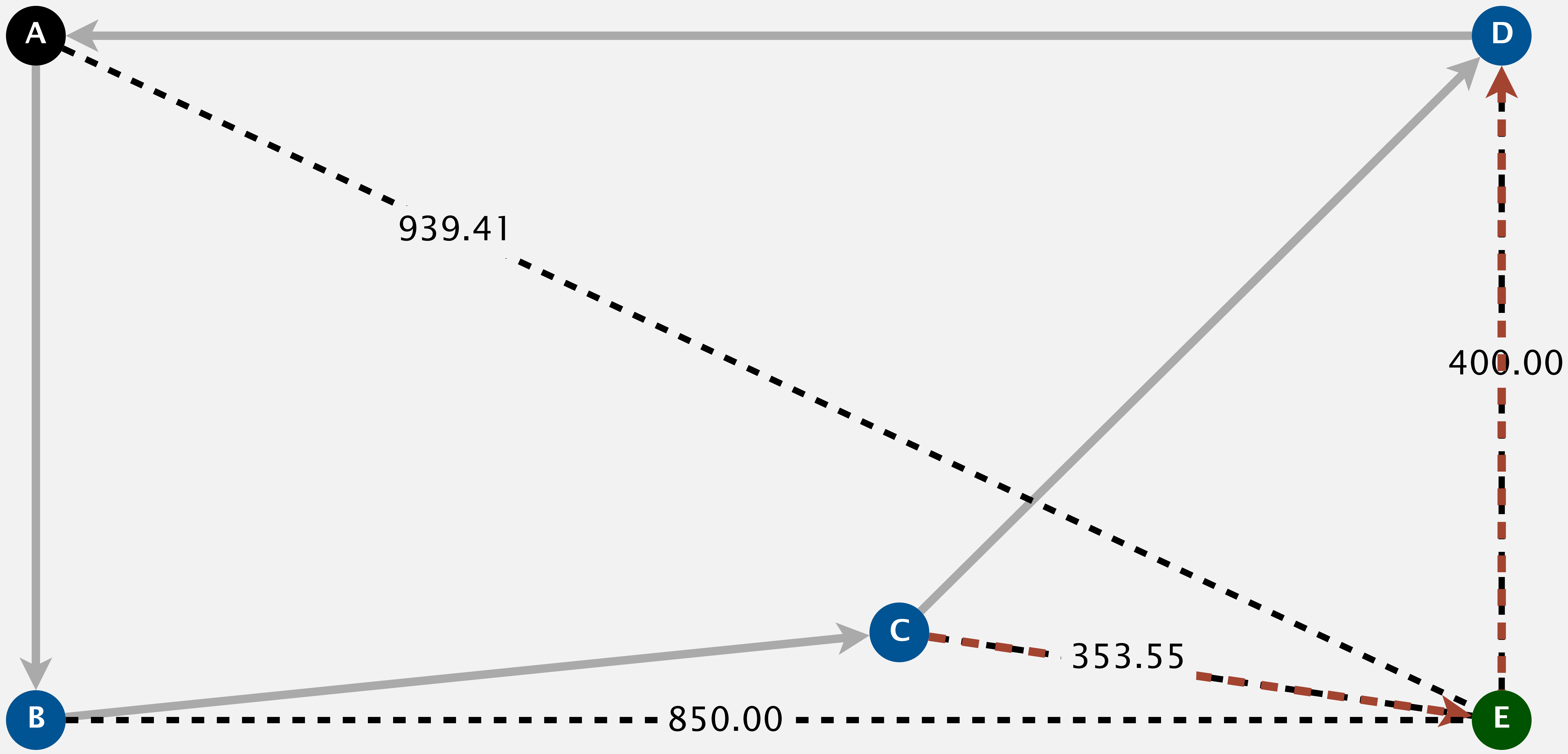
Assignment 8 Tips and Tricks

- traveling salesperson problem
- nearest insertion heuristic
- smallest increase heuristic
- implementation
- beyond

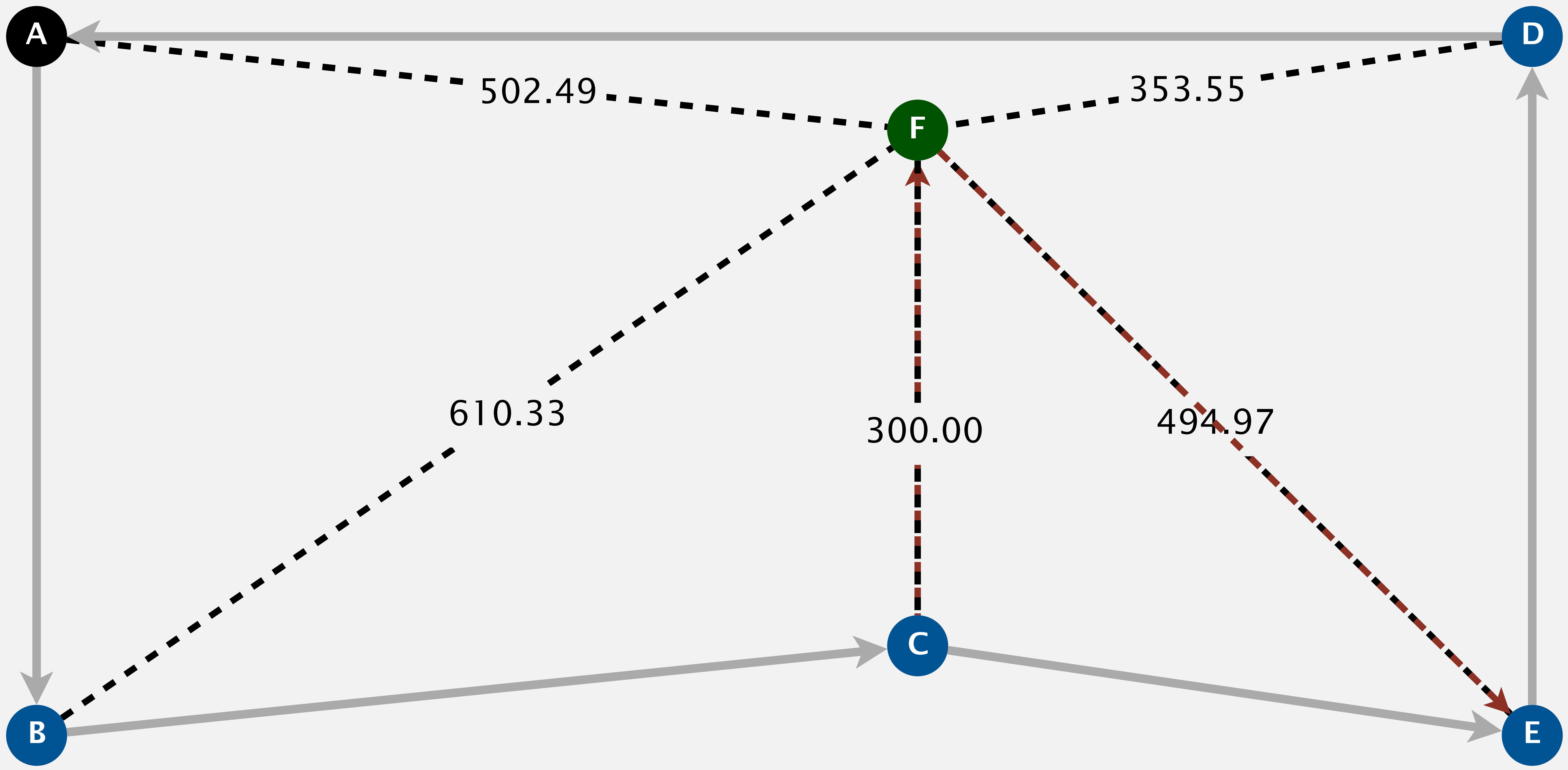
Nearest insertion heuristic



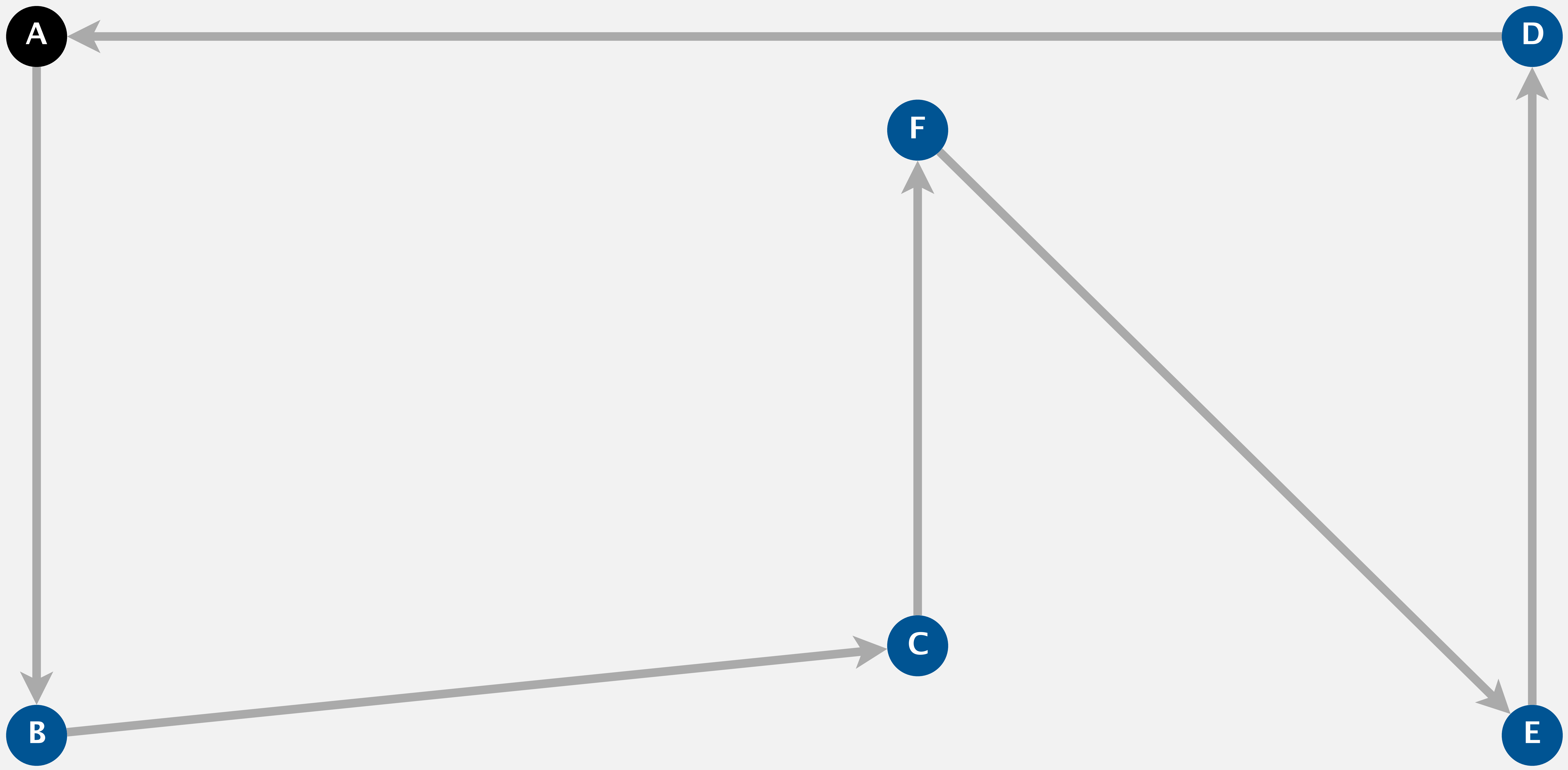
Nearest insertion heuristic



Nearest insertion heuristic



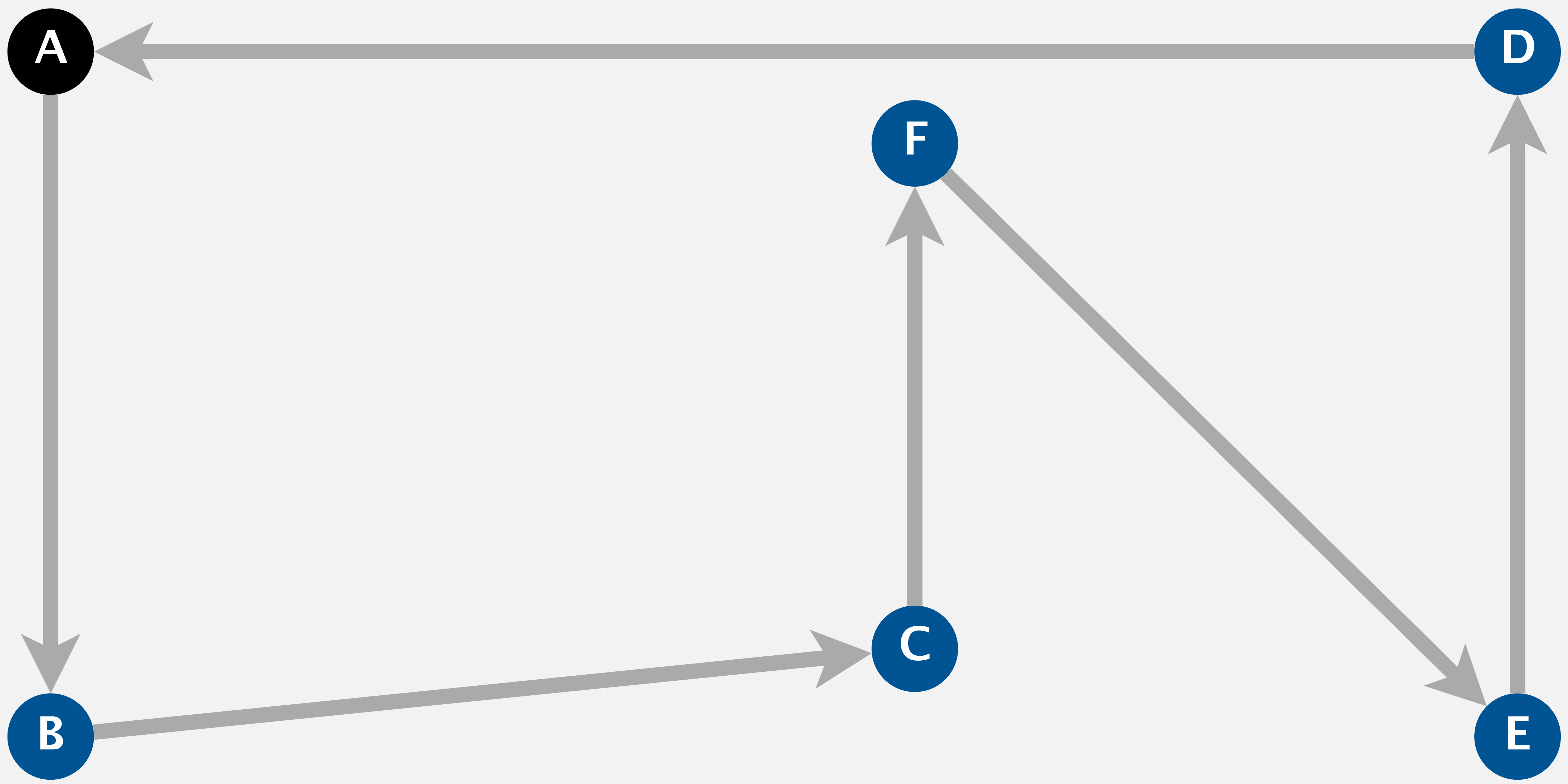
Nearest insertion heuristic



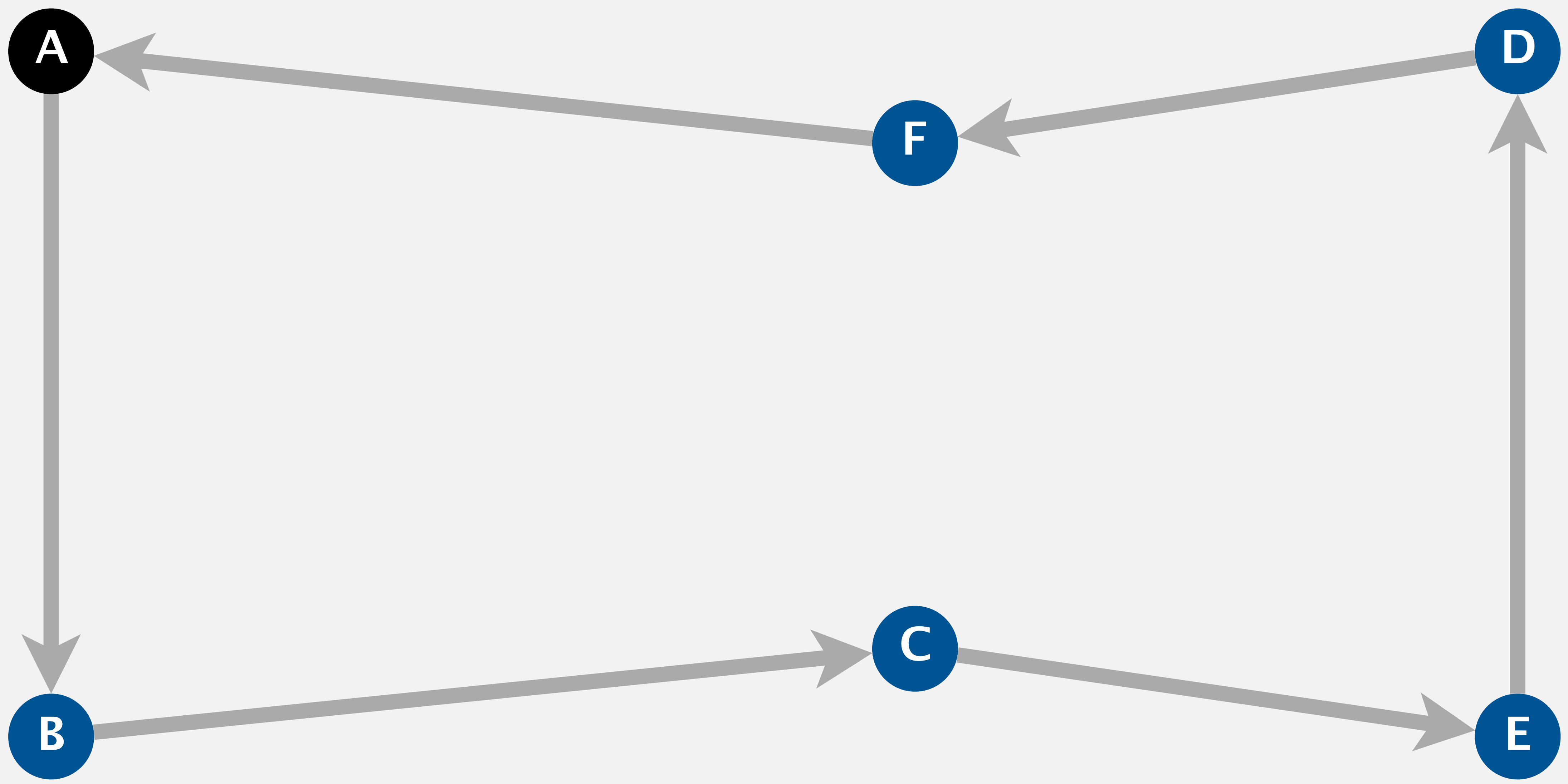
Nearest insertion heuristic

Q. Does nearest insertion heuristic guarantee to produce shortest tour?

A. No.



nearest insertion tour length = 2947.47

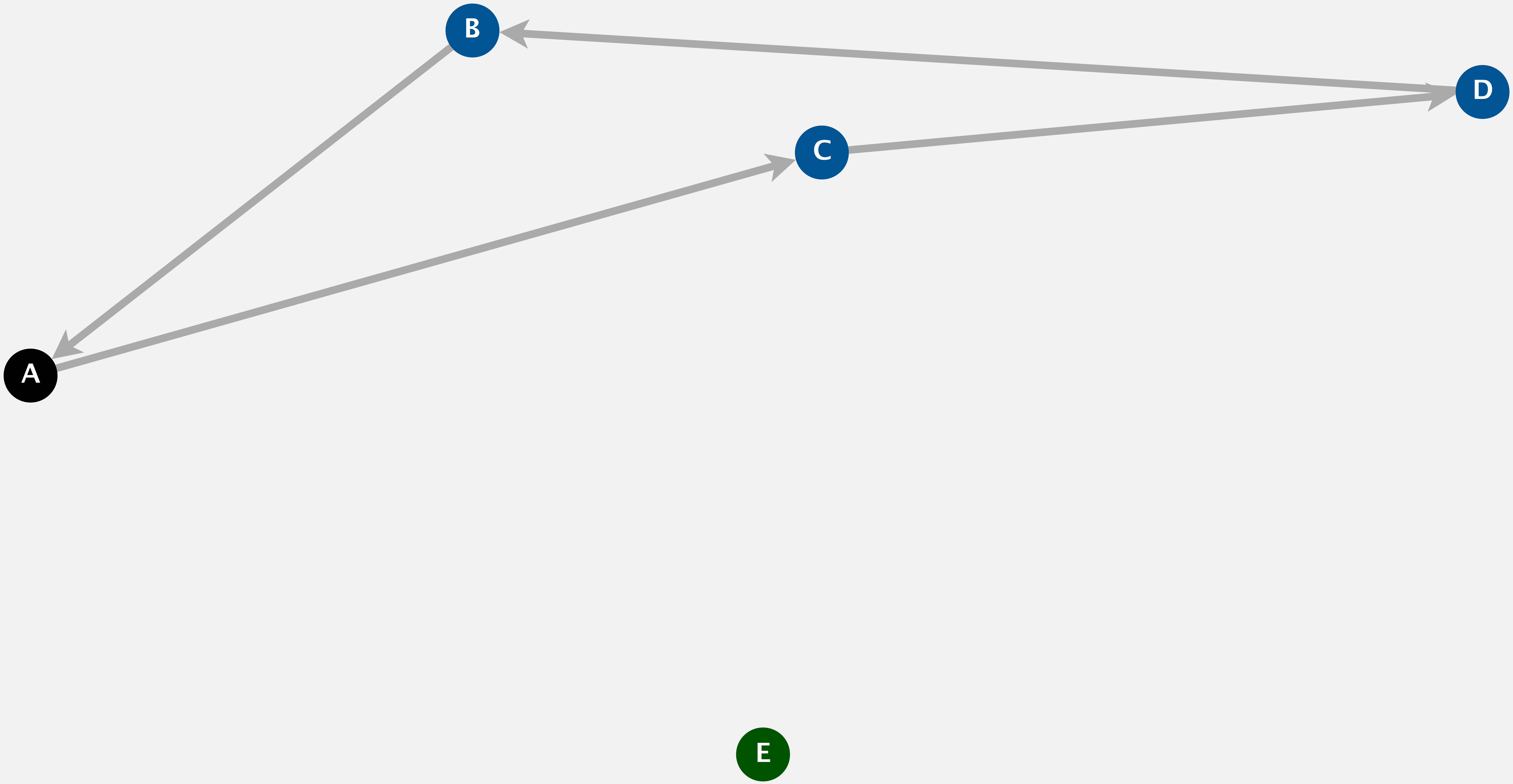


optimal tour length = 2512.09

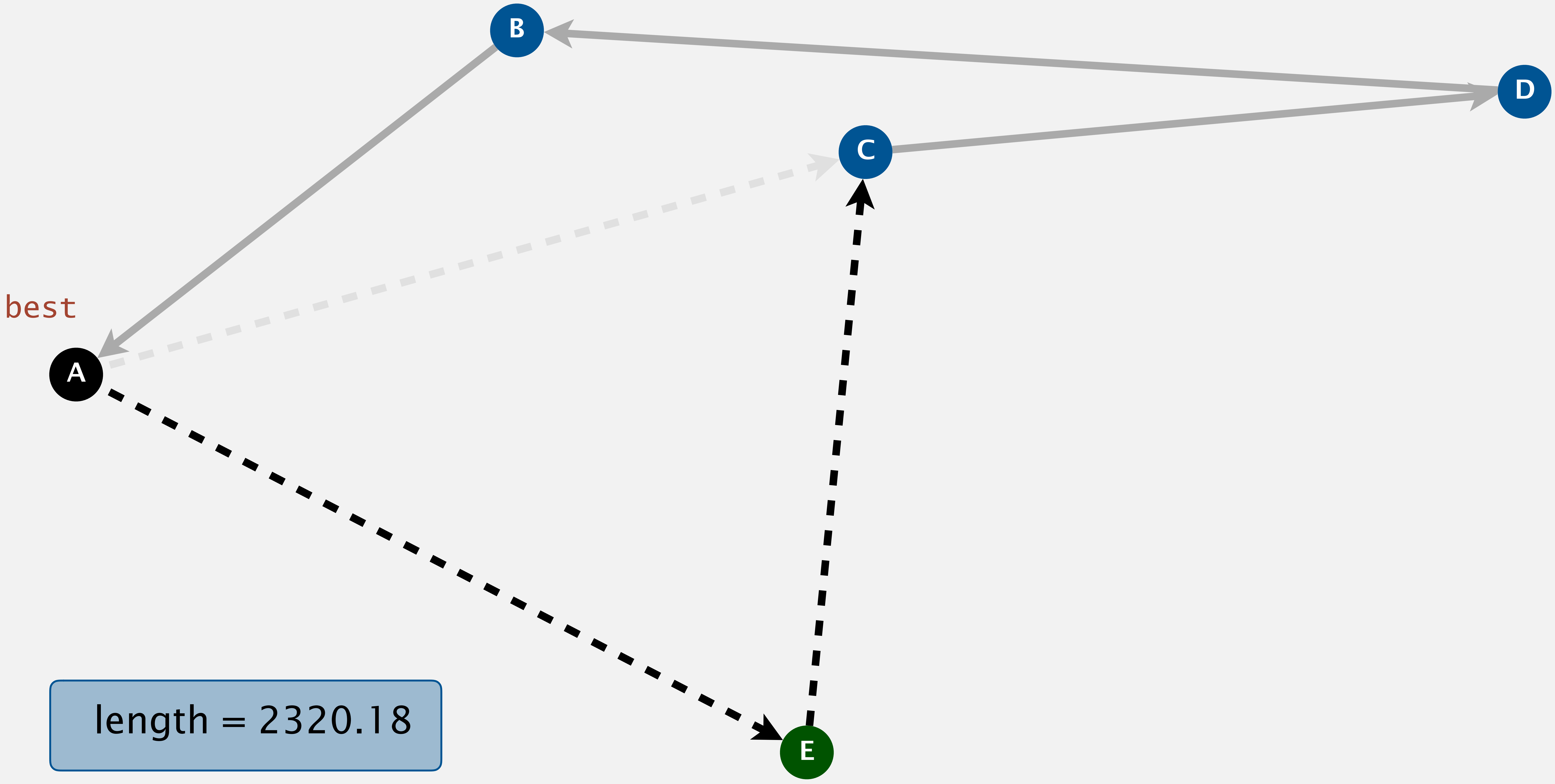
Assignment 8 Tips and Tricks

- *traveling salesperson problem*
- *nearest insertion heuristic*
- *smallest increase heuristic*
- *implementation*
- *beyond*

Smallest increase heuristic

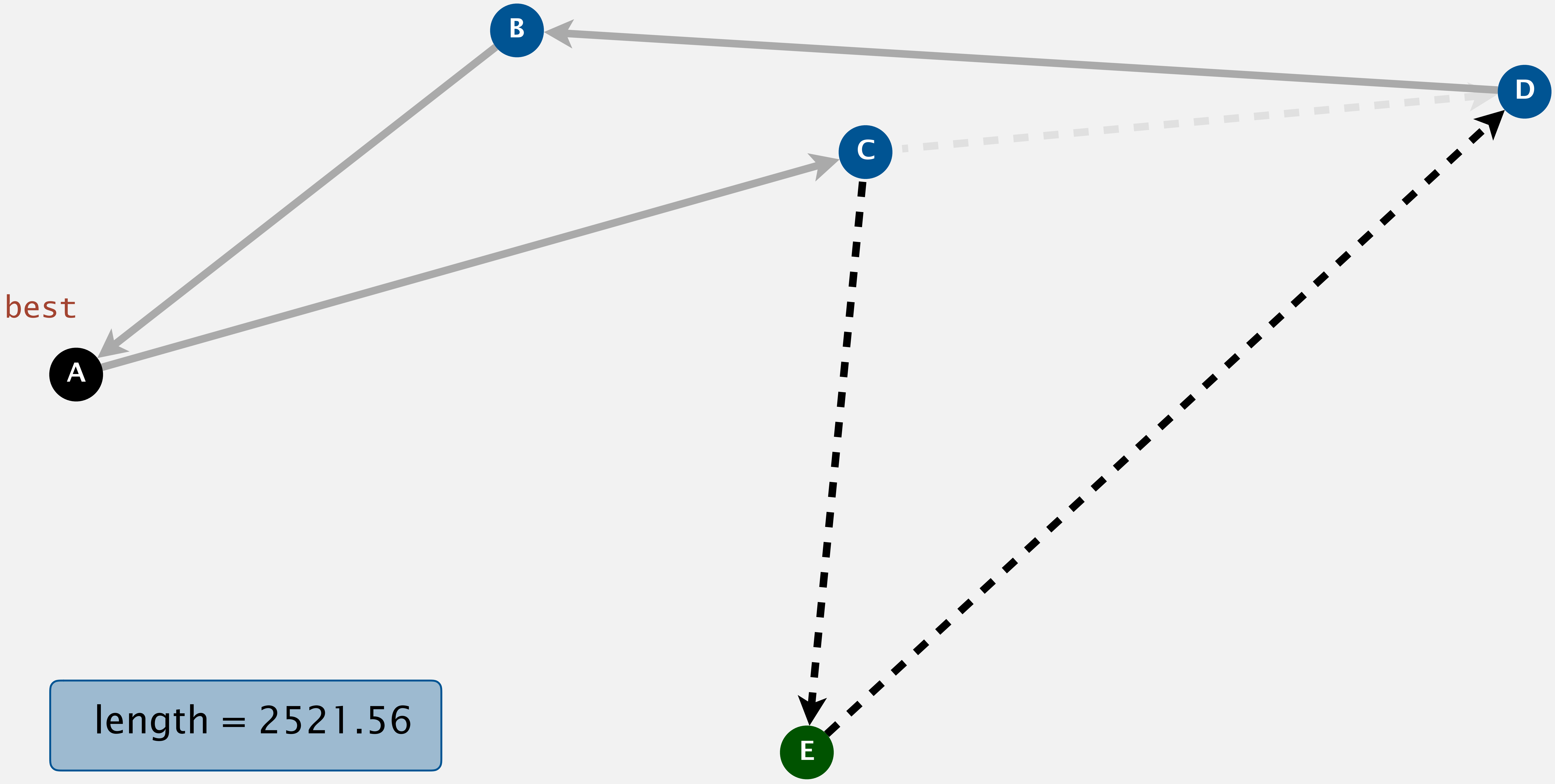


Smallest increase heuristic



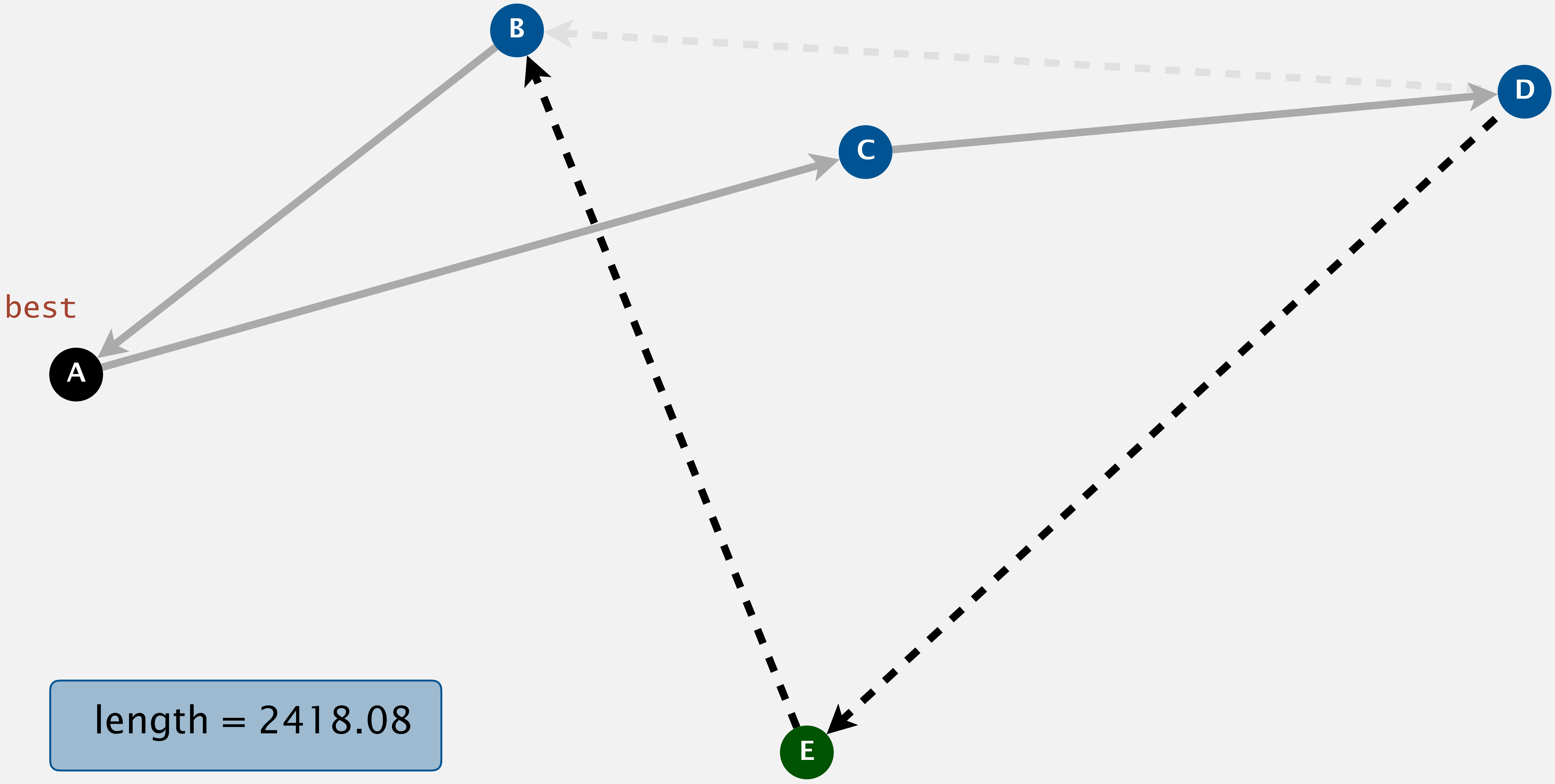
length = 2320.18

Smallest increase heuristic



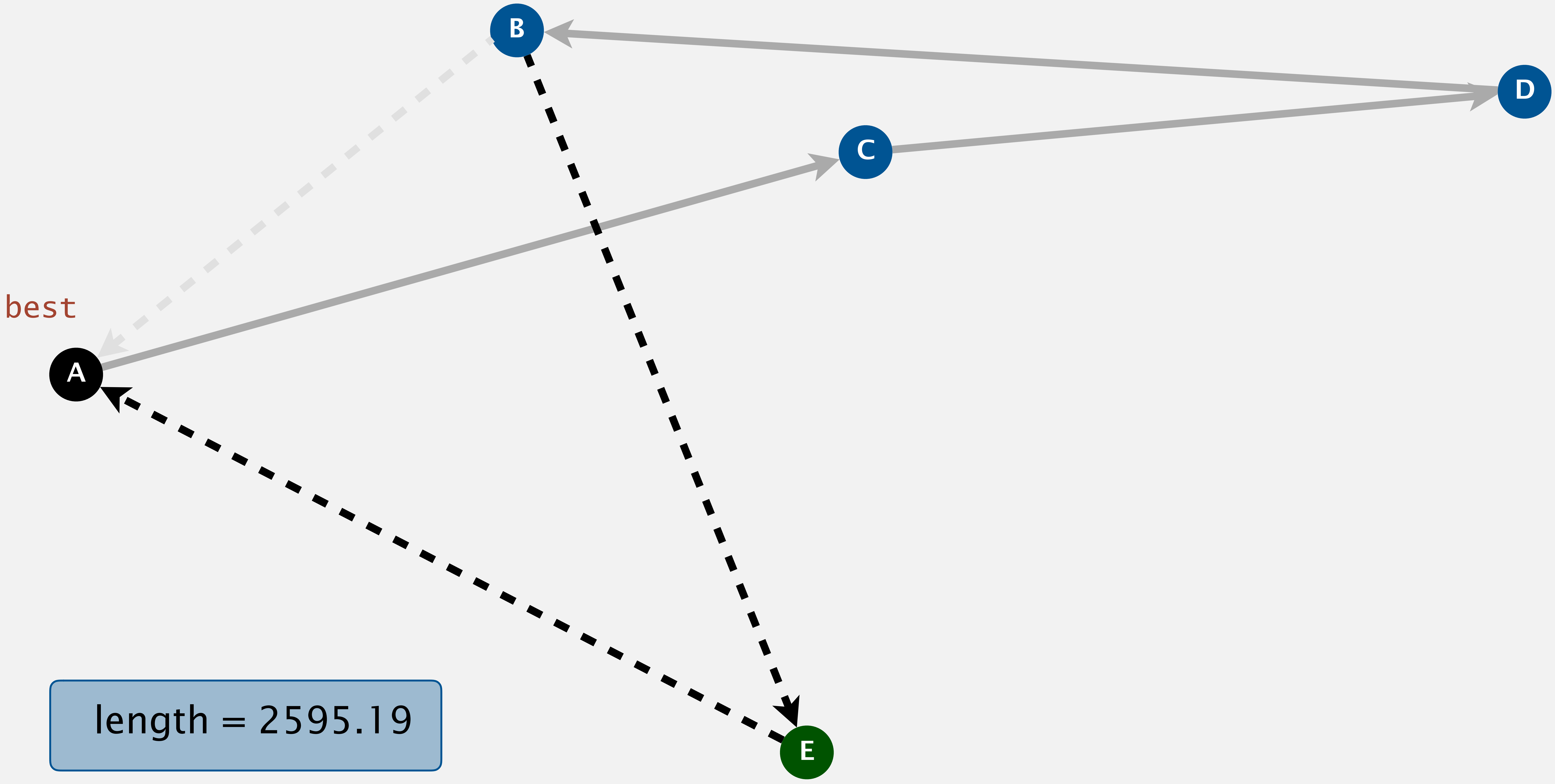
length = 2521.56

Smallest increase heuristic



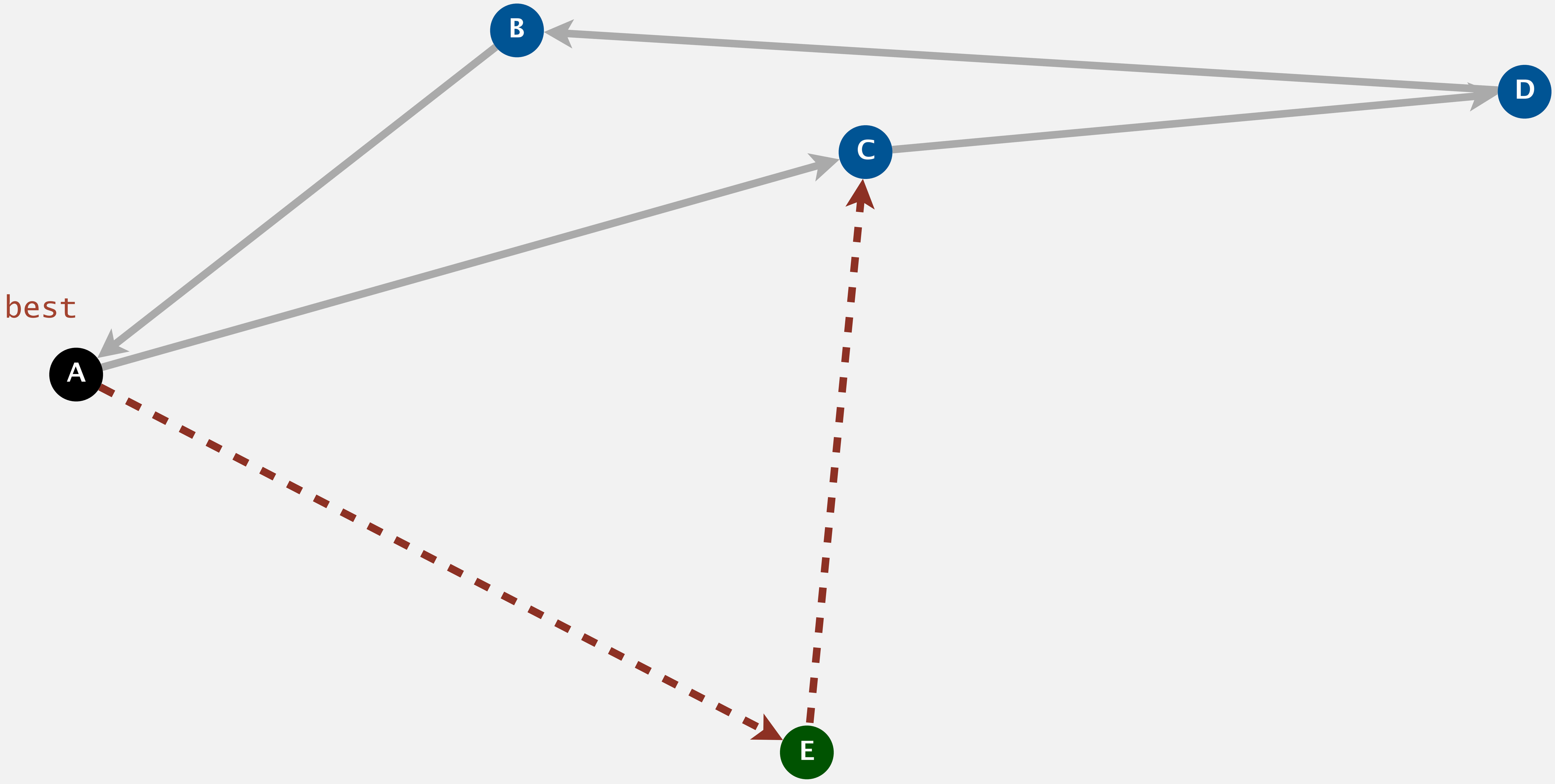
length = 2418.08

Smallest increase heuristic

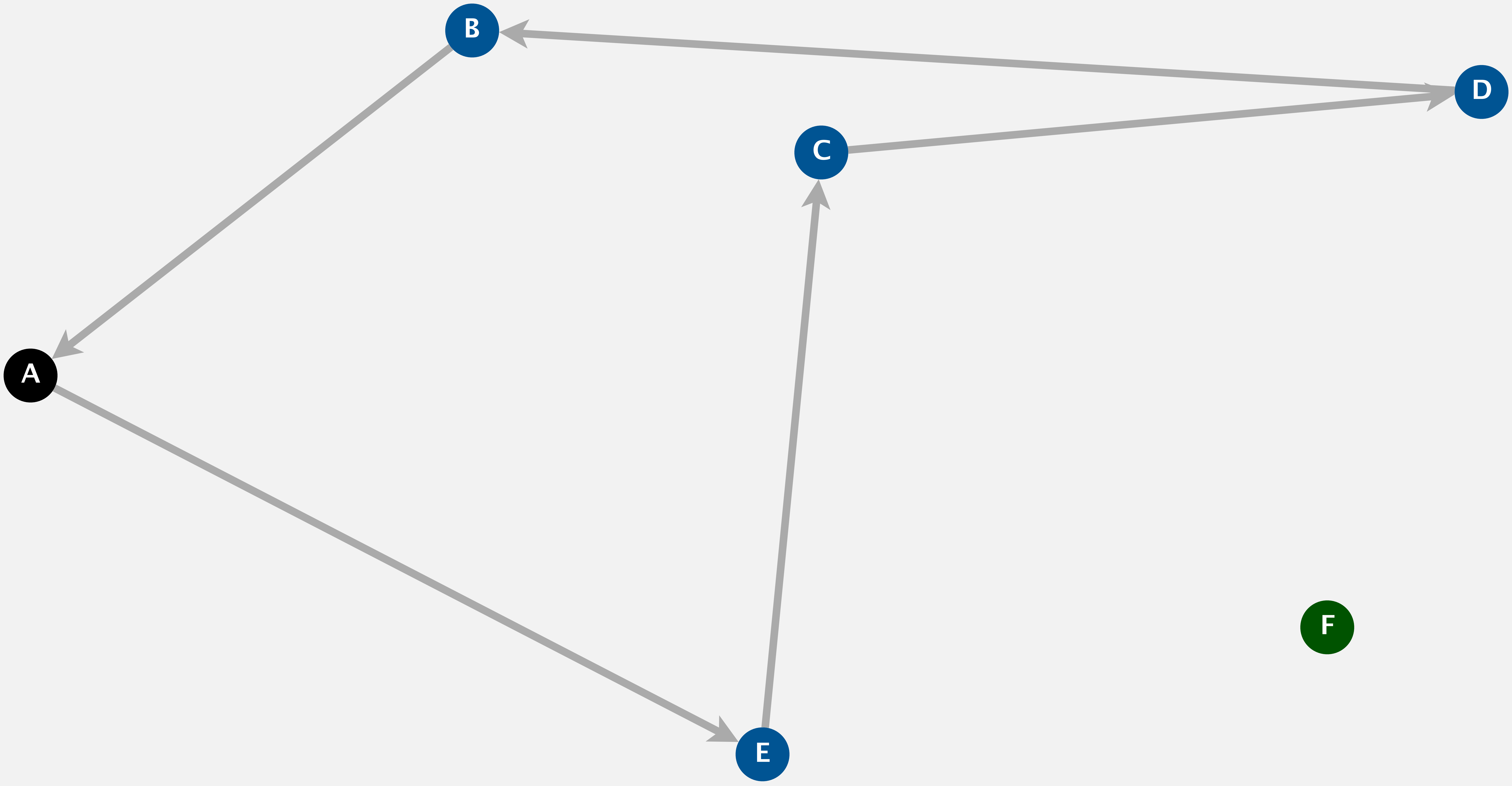


length = 2595.19

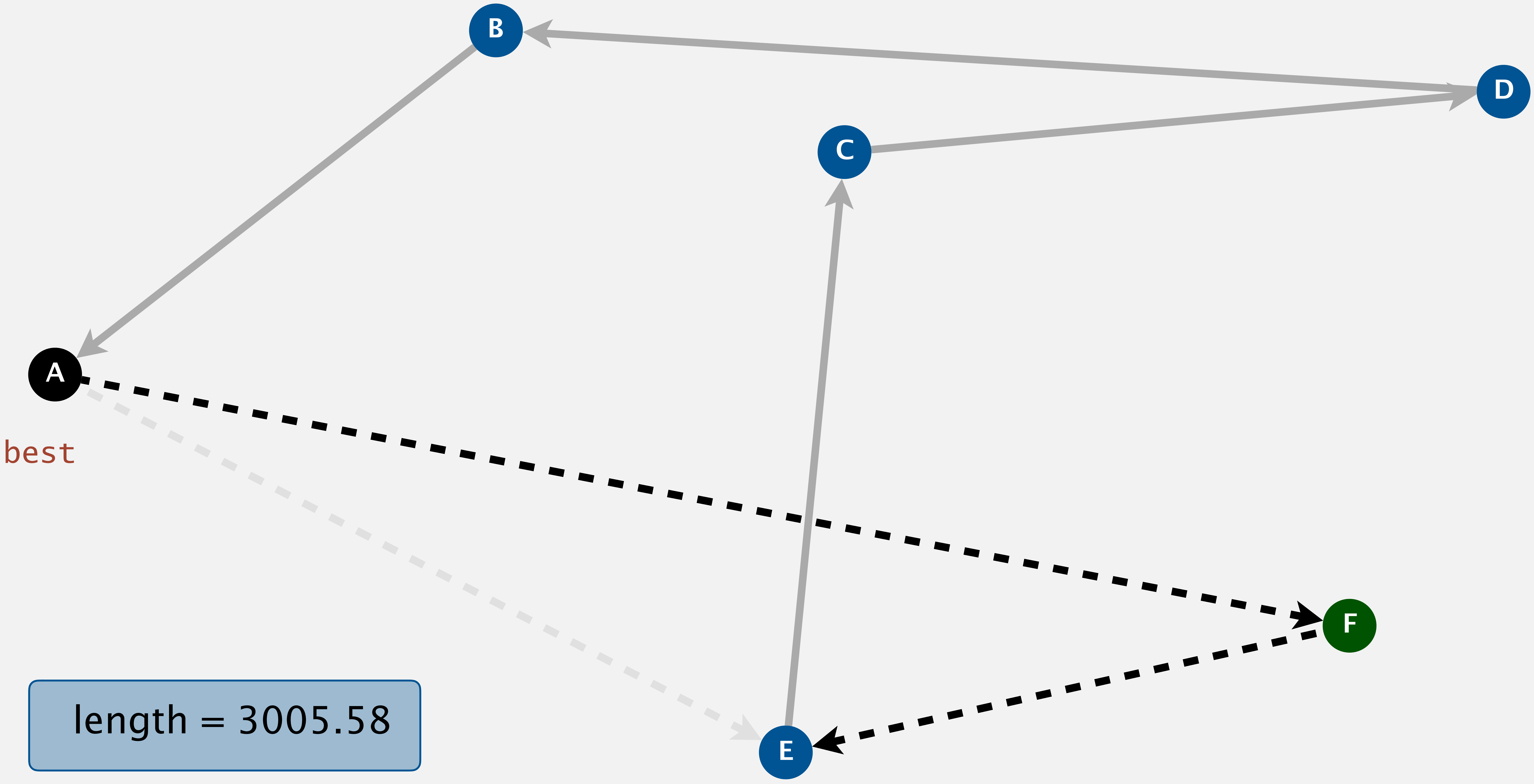
Smallest increase heuristic



Smallest increase heuristic

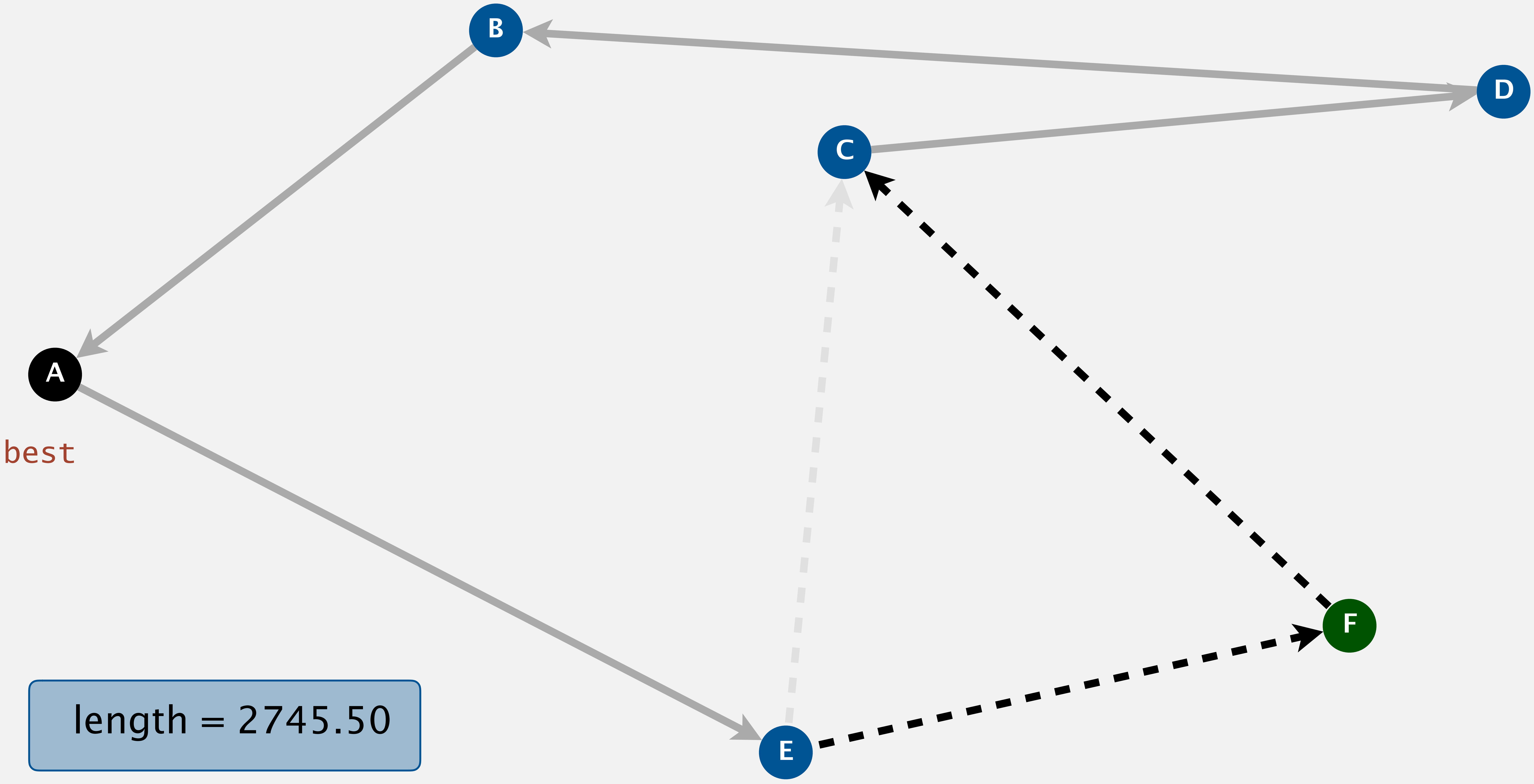


Smallest increase heuristic



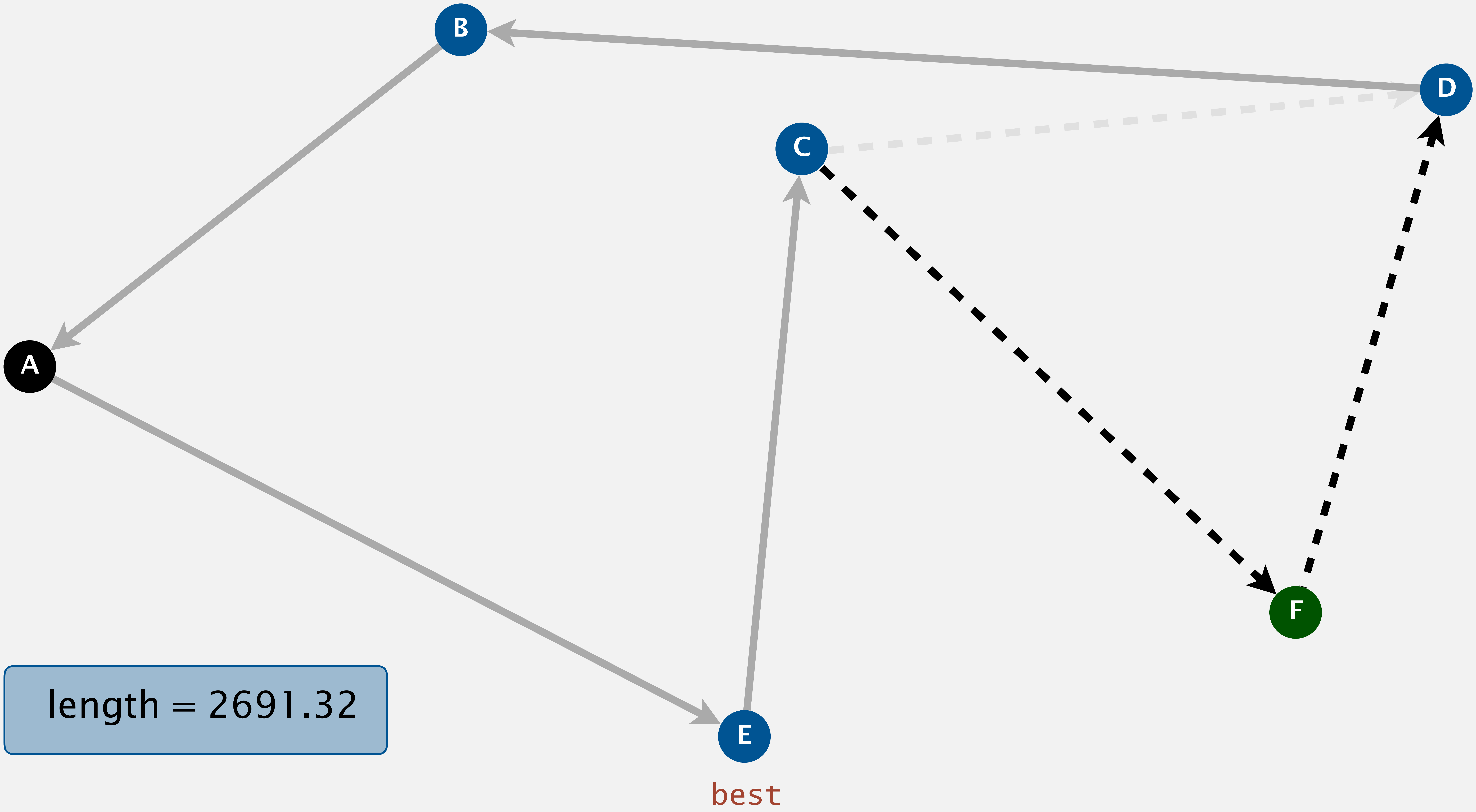
length = 3005.58

Smallest increase heuristic

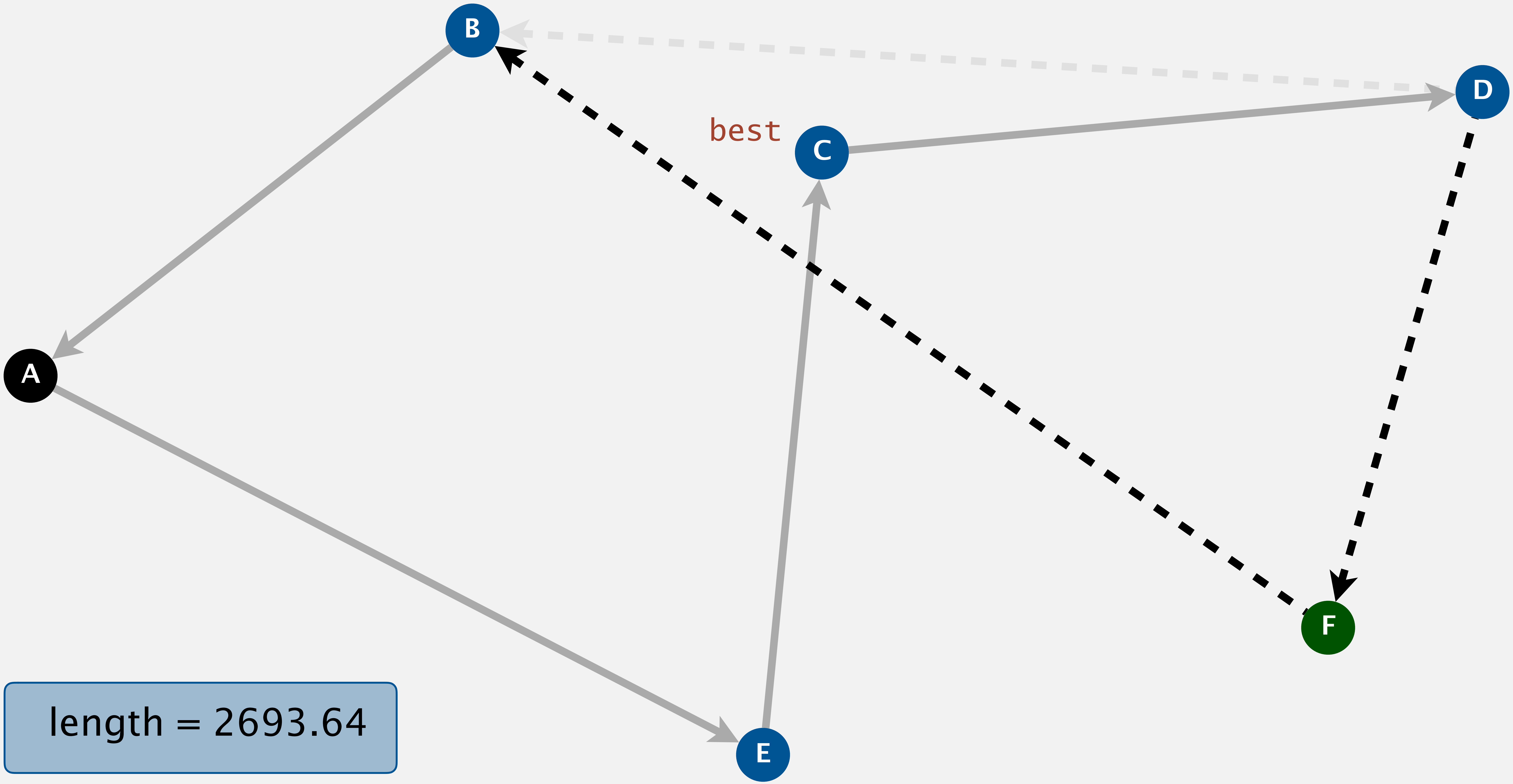


length = 2745.50

Smallest increase heuristic

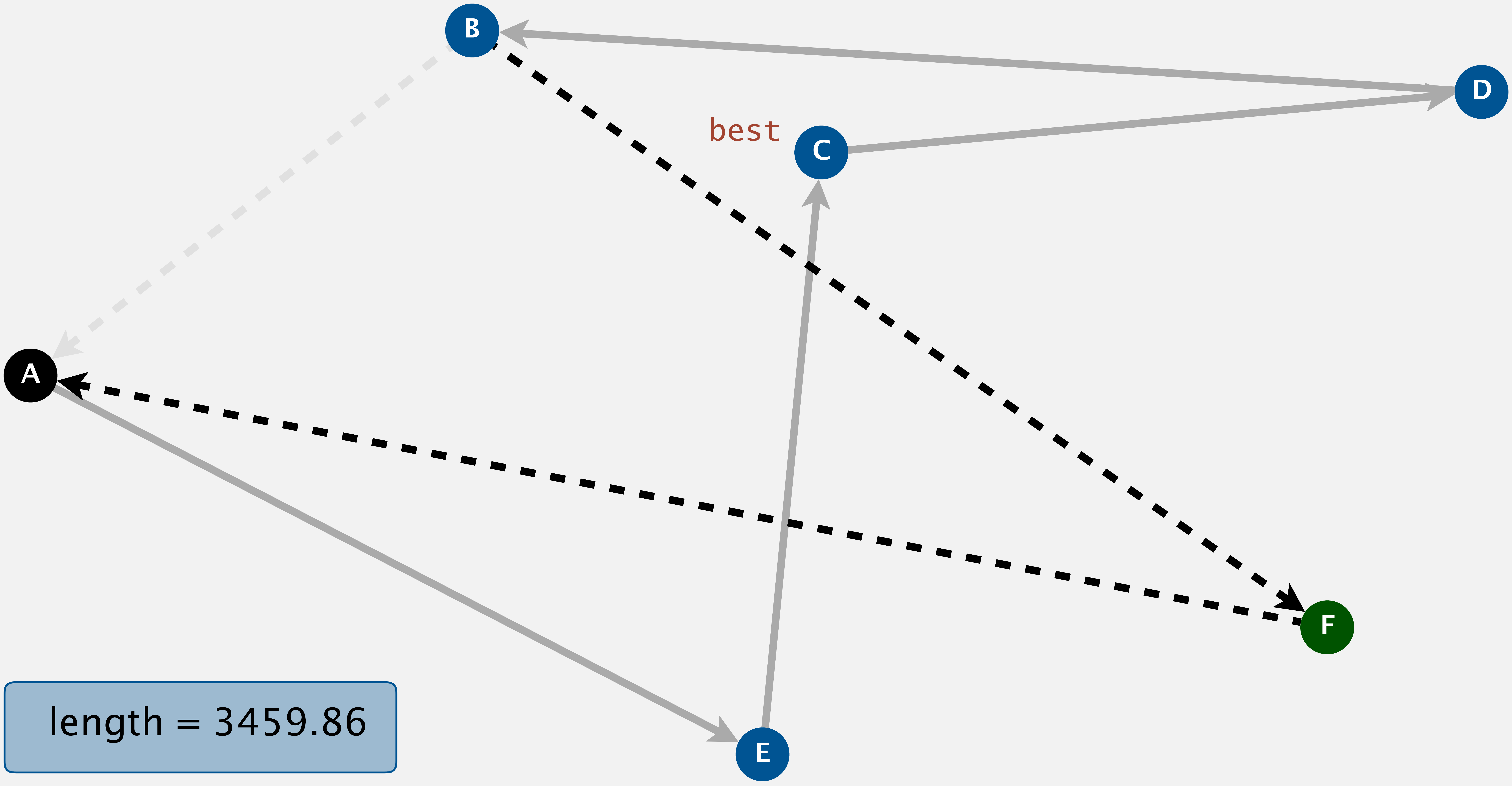


Smallest increase heuristic



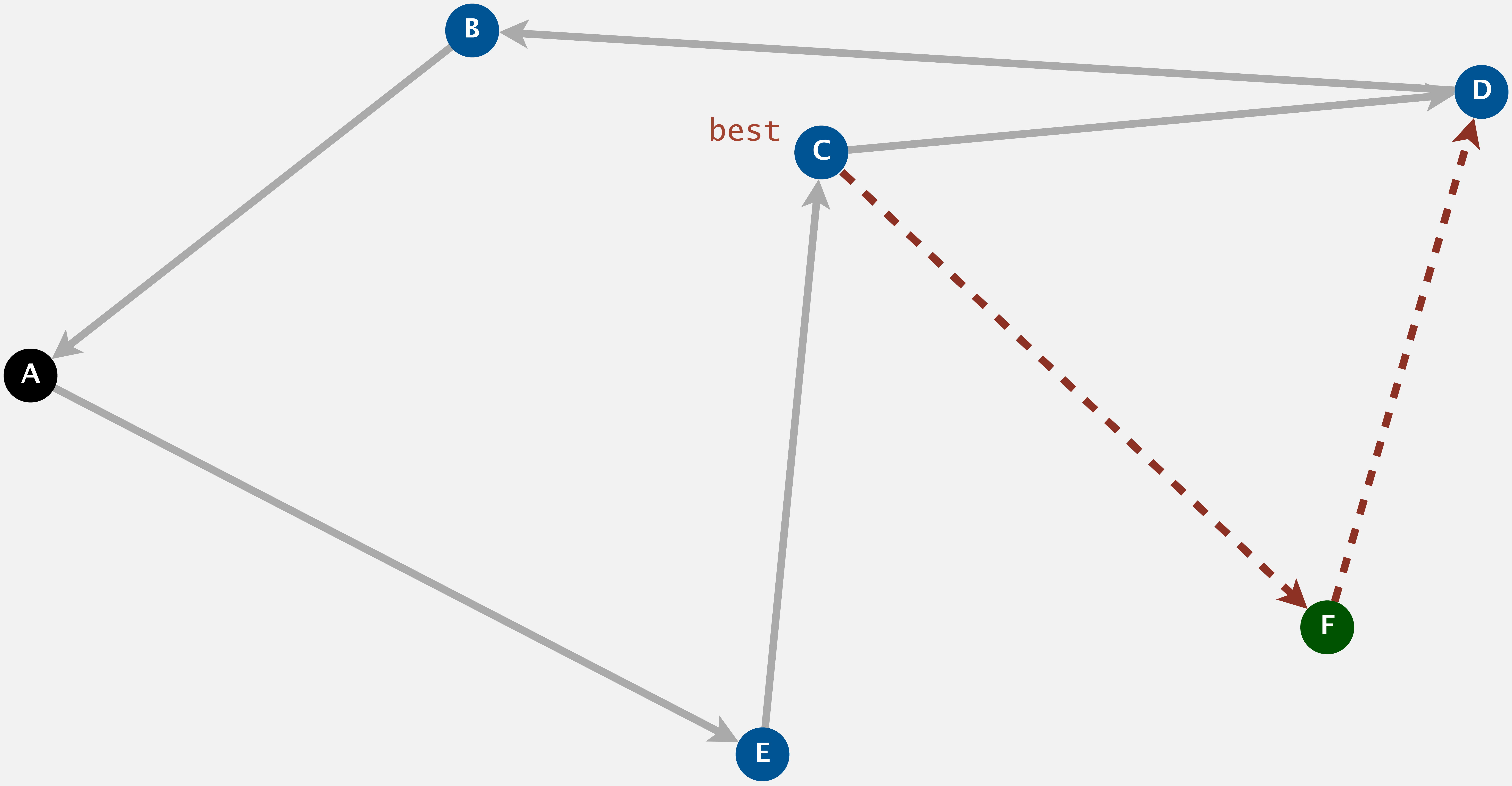
length = 2693.64

Smallest increase heuristic

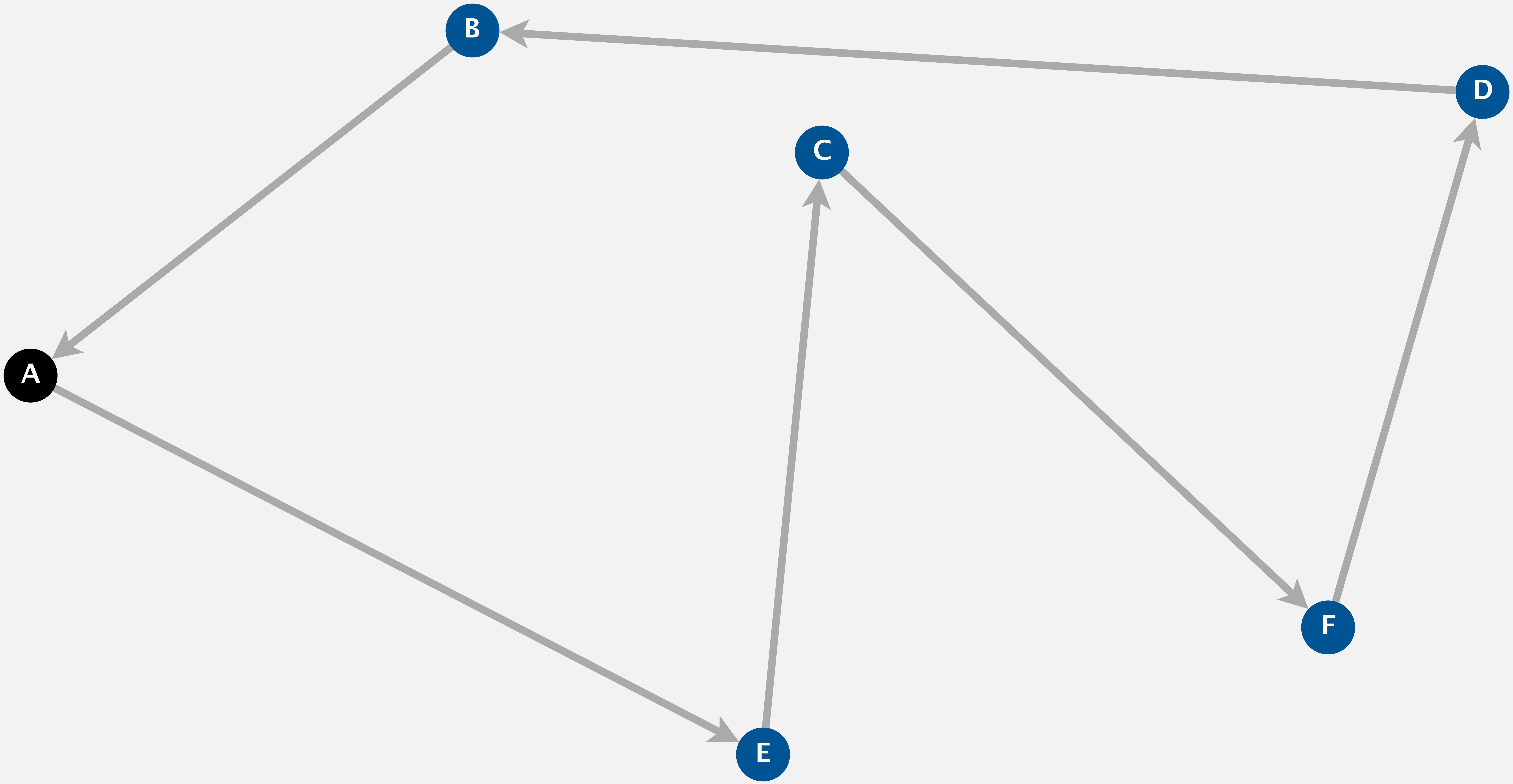


length = 3459.86

Smallest increase heuristic



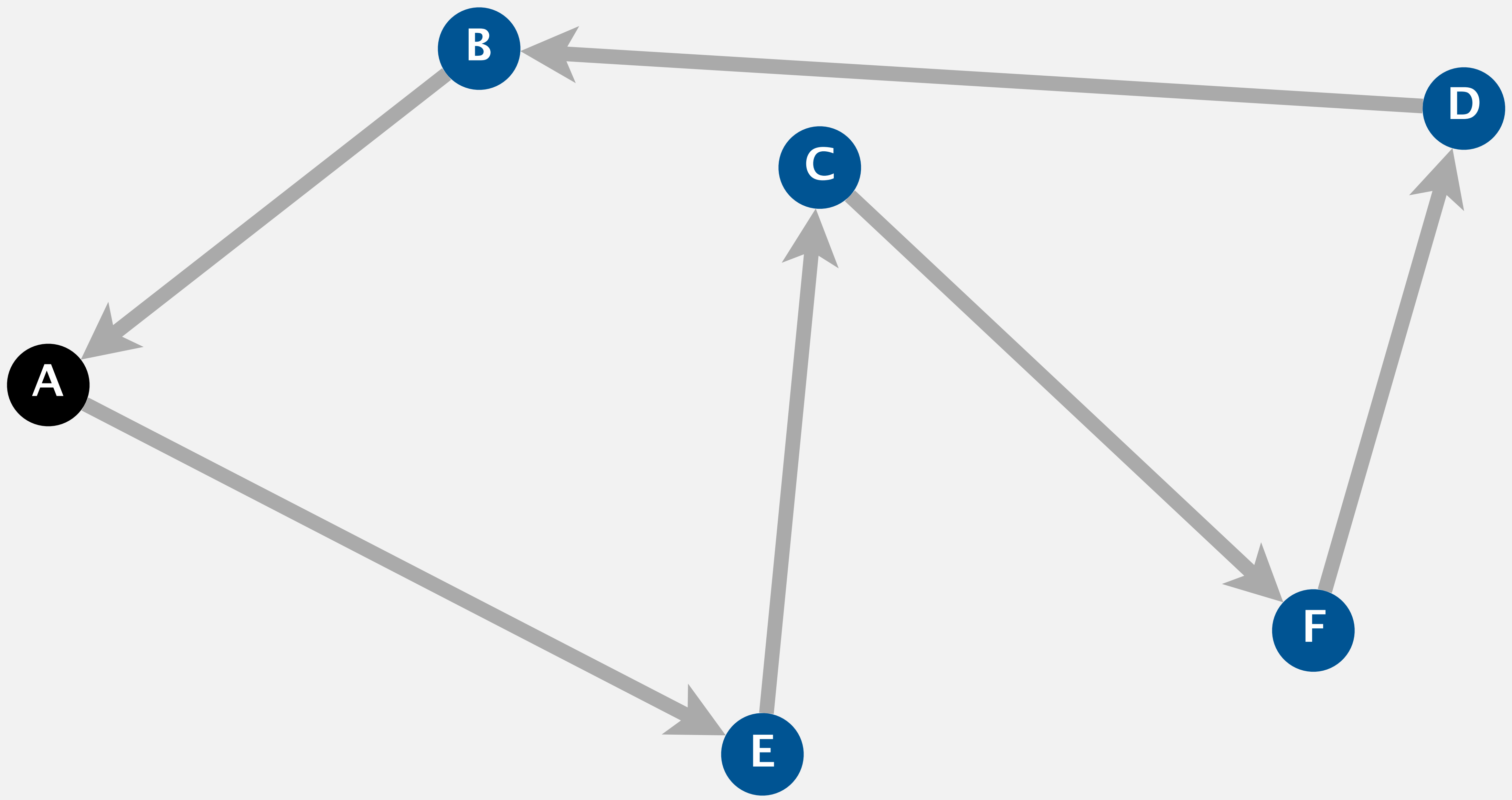
Smallest increase heuristic



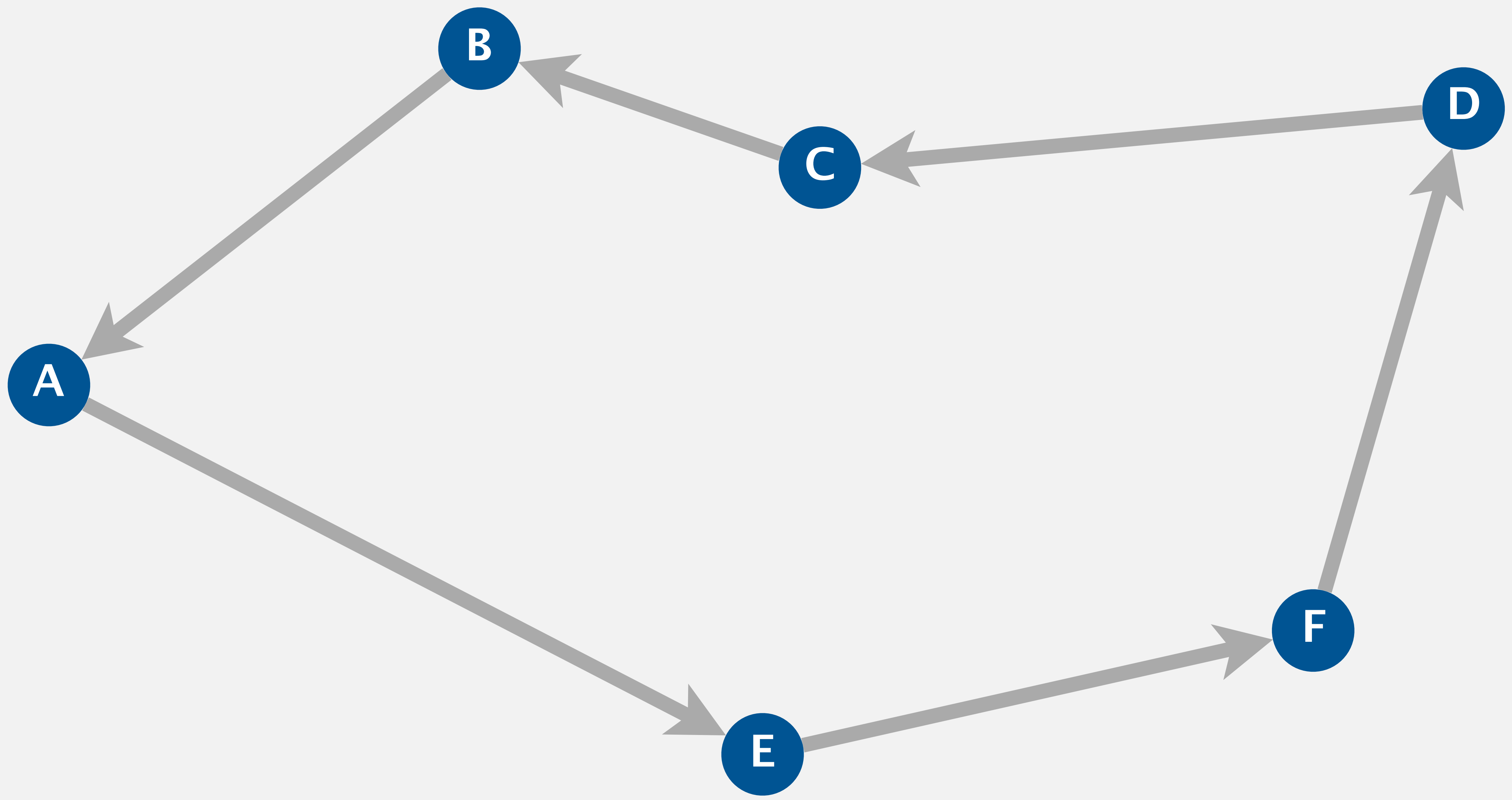
Smallest increase heuristic

Q. Does smallest increase heuristic guarantee to produce shortest tour?

A. No.



smallest increase tour length = 2691.31



optimal tour length = 2254.11

Assignment 8 Tips and Tricks

- *traveling salesperson problem*
- *nearest insertion heuristic*
- *smallest increase heuristic*
- *implementation*
- *beyond*

Point data type

You will not write or submit this file.

```
public   class Point
```

	`public Point(double x, double y)`	*creates the point (x, y)*
`public`	`double distanceTo(Point that)`	*Euclidean distance between the two points*
`public`	`int drawTo(Point that)`	*draws the line segment between the two points*
`public`	`String toString()`	*string representation of this point*

Tour data type

`public class Tour`

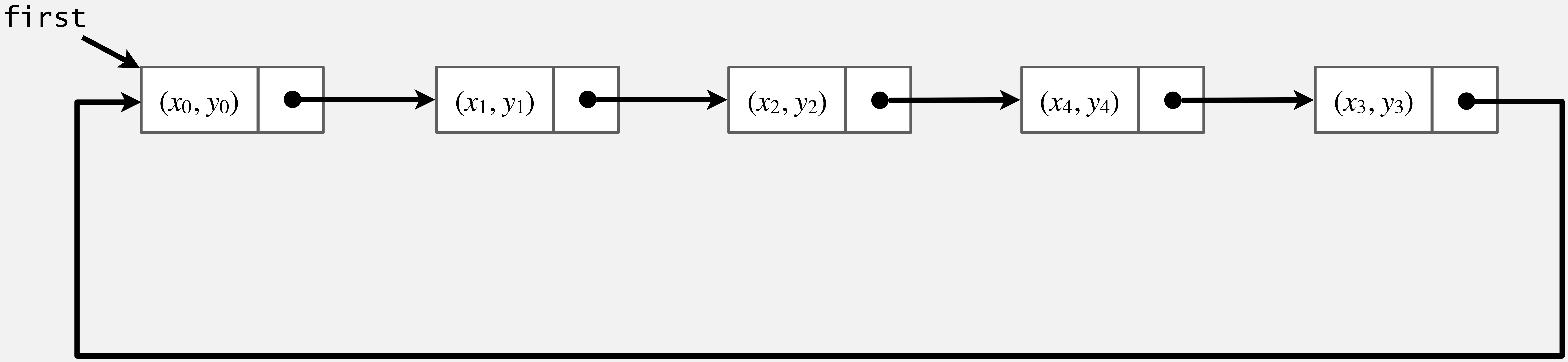
`public`	`Tour()`	*creates an empty tour*
`public`	`Tour(Point a, Point b, … )`	*creates a 4-point tour a→b→c→d→a*
`public int`	`size()`	*number of points in this tour*
`public double`	`length()`	*length of this tour*
`public void`	`draw()`	*draws this tour to standard drawing*
`public String`	`toString()`	*string representation of this tour*
`public void`	`insertNearest(Point p)`	*inserts the point p into tour using nearest insertion heuristic*
`public void`	`insertSmallest(Point p)`	*inserts the point p into the tour using the smallest increase heuristic*

Circularly linked lists

Node data type.

```
private class Node {
    private Point p;
    private Node next;
}
```

Visual representation.



Traversing a circularly linked list

Which of the following prints every node in a circularly linked list?

A.

```
Node x = first;
while (x != first) {
    StdOut.println(x.p);
    first = first.next;
}
```

B.

```
Node x = first;
while (x.next != first) {
    StdOut.println(x.p);
    x = x.next;
}
```

C.

```
for (Node x = first; x != null; x = x.next) {
    StdOut.println(x.p);
}
```

D.

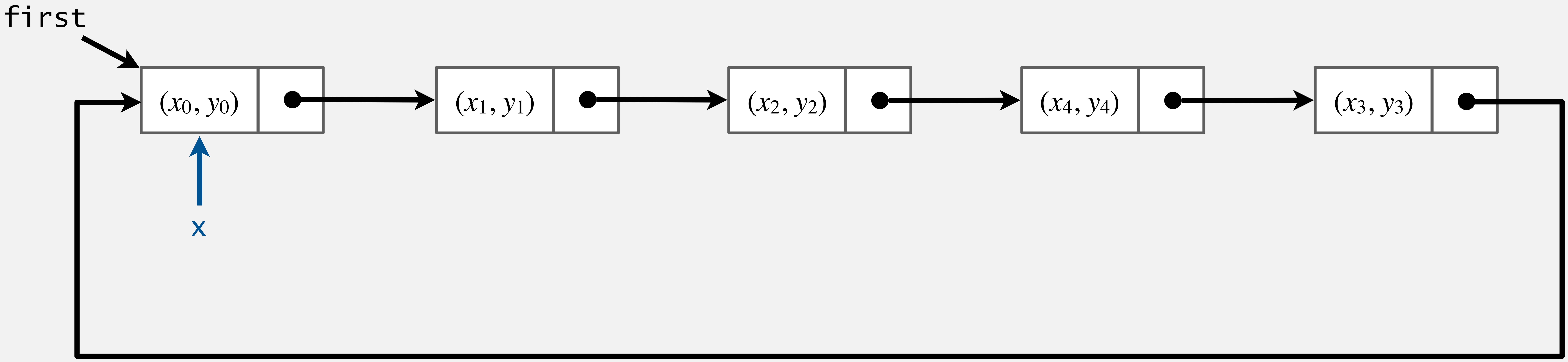
```
Node x = first;
do {
    StdOut.println(x.p);
    x = x.next;
} while (x != first);
```

Traversing a circularly linked list

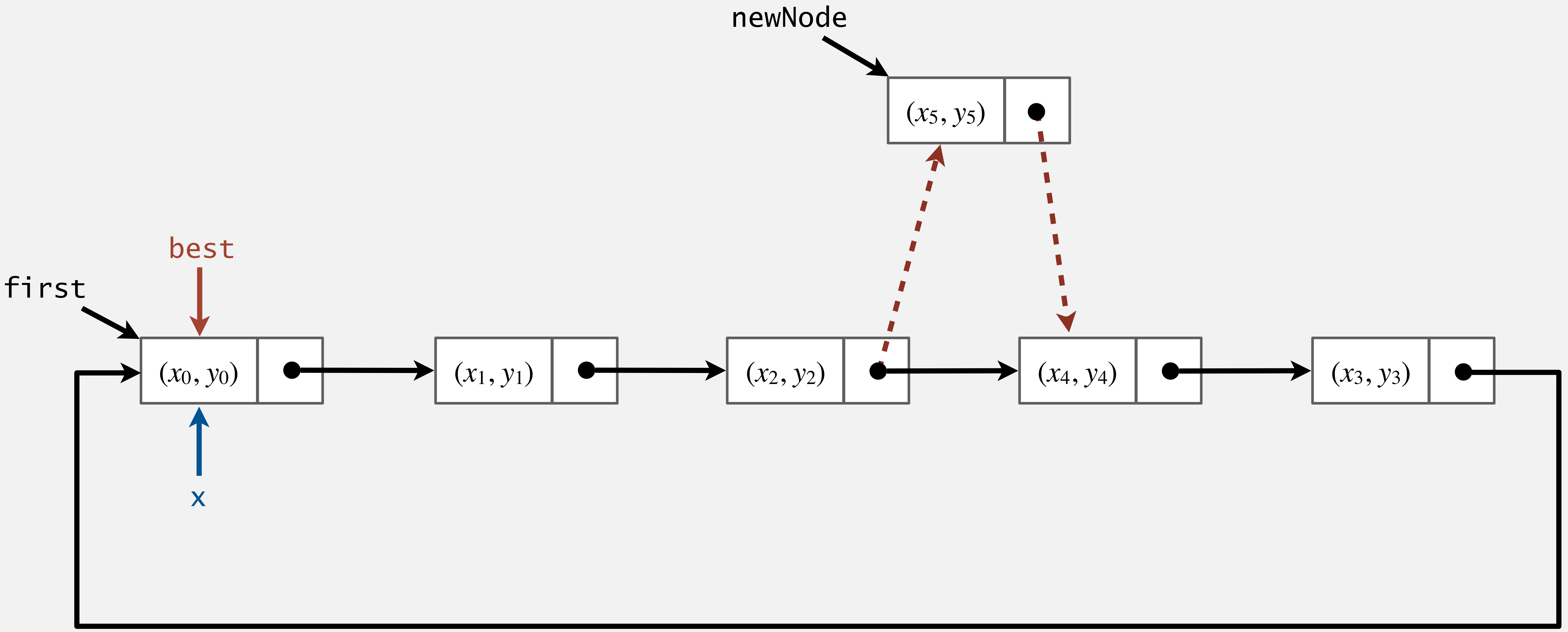
```
Node x = first;
do {
    StdOut.println(x.p);
    x = x.next;
} while (x != first);
```

standard output

(x_0, y_0)
(x_1, y_1)
(x_2, y_2)
(x_4, y_4)
(x_3, y_3)



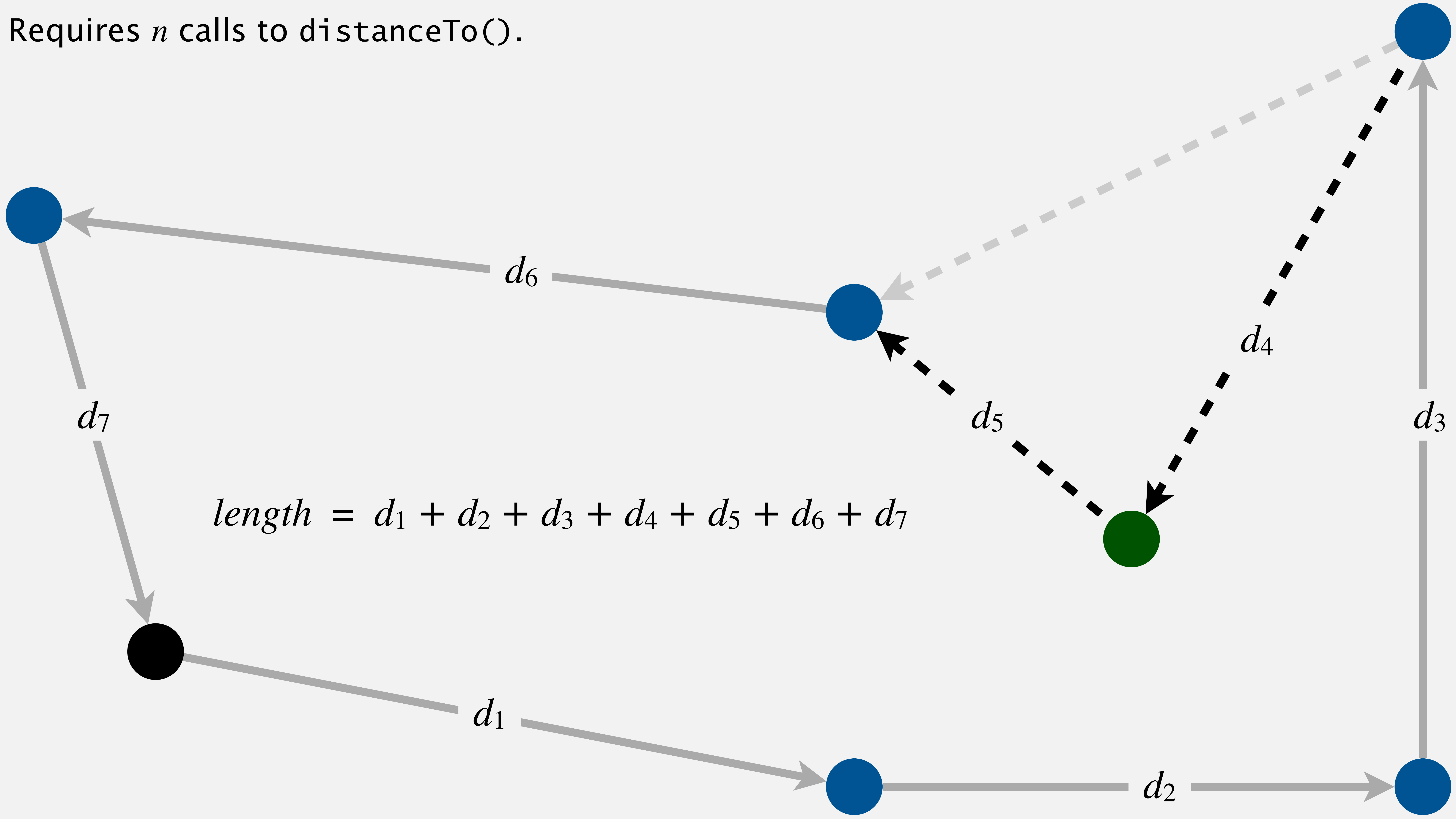
Inserting a node into a circularly linked list



Smallest increase heuristic: performance trick

Bottleneck. Computing the tour length.

Impact. Requires n calls to `distanceTo()`.

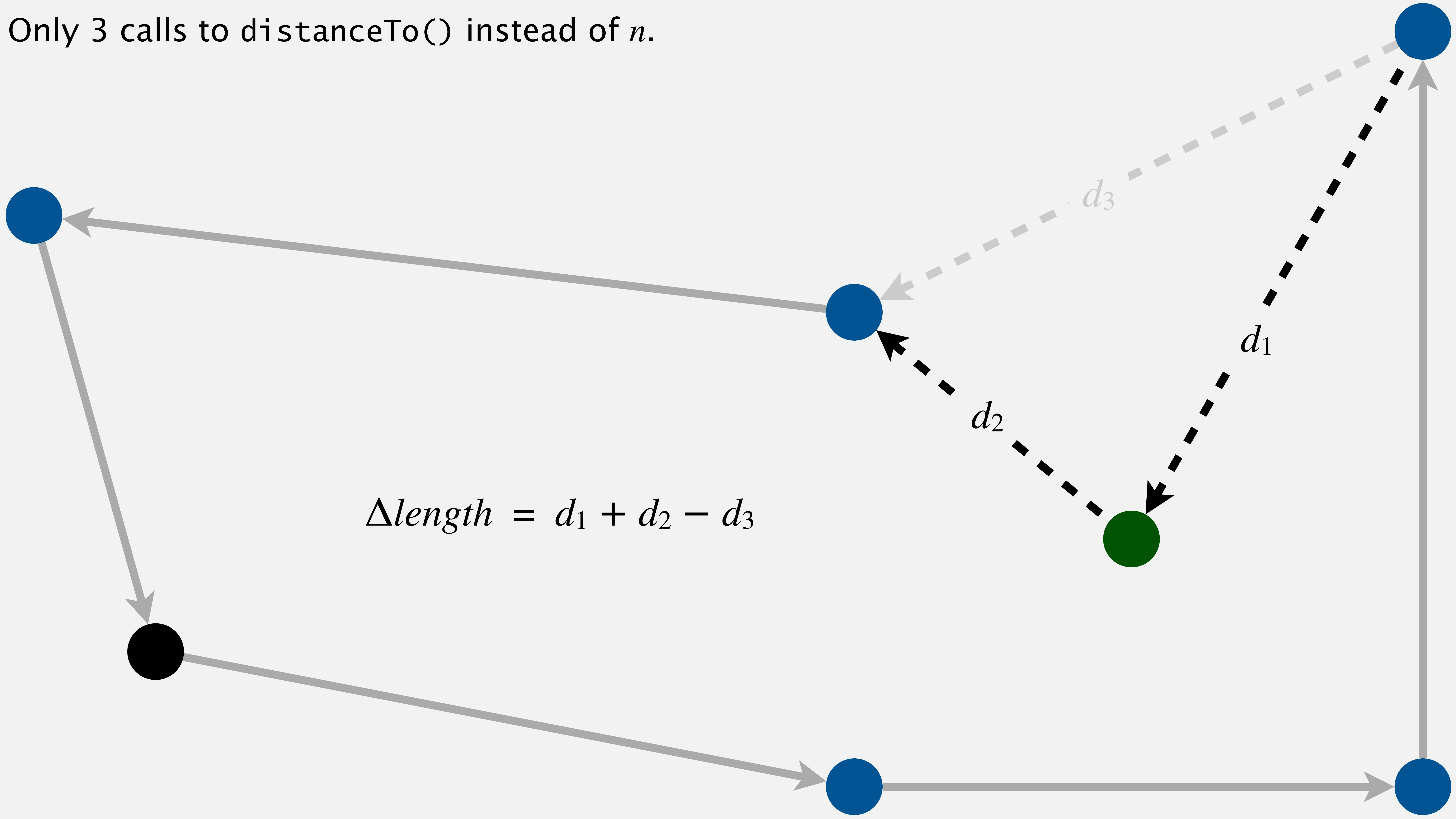


$$length = d_1 + d_2 + d_3 + d_4 + d_5 + d_6 + d_7$$

Smallest increase heuristic: performance trick

Key optimization. Compute change in tour length; not tour length.

Impact. Only 3 calls to `distanceTo()` instead of n.



$$\Delta length = d_1 + d_2 - d_3$$

Tips and tricks

Linked structures.

- Do not use a null-terminated linked list.
- You must use a circularly linked list.
- Use `new Node()` only to create new nodes (not new references to existing nodes).

Traversing the circularly linked list.

- Can use a `for` or `while` loop.
- Simpler with a `do-while` loop. ← see `CircularQuote` in precept

Work incrementally. Constructors, `size()`, `length()`, `toString()`, ...

Dealing with ties. If tie, insert after first such point.

Corner cases. (0- and 1-point tours).

String representation. Use `StringBuilder` in `toString()` for repeated string concatenation.

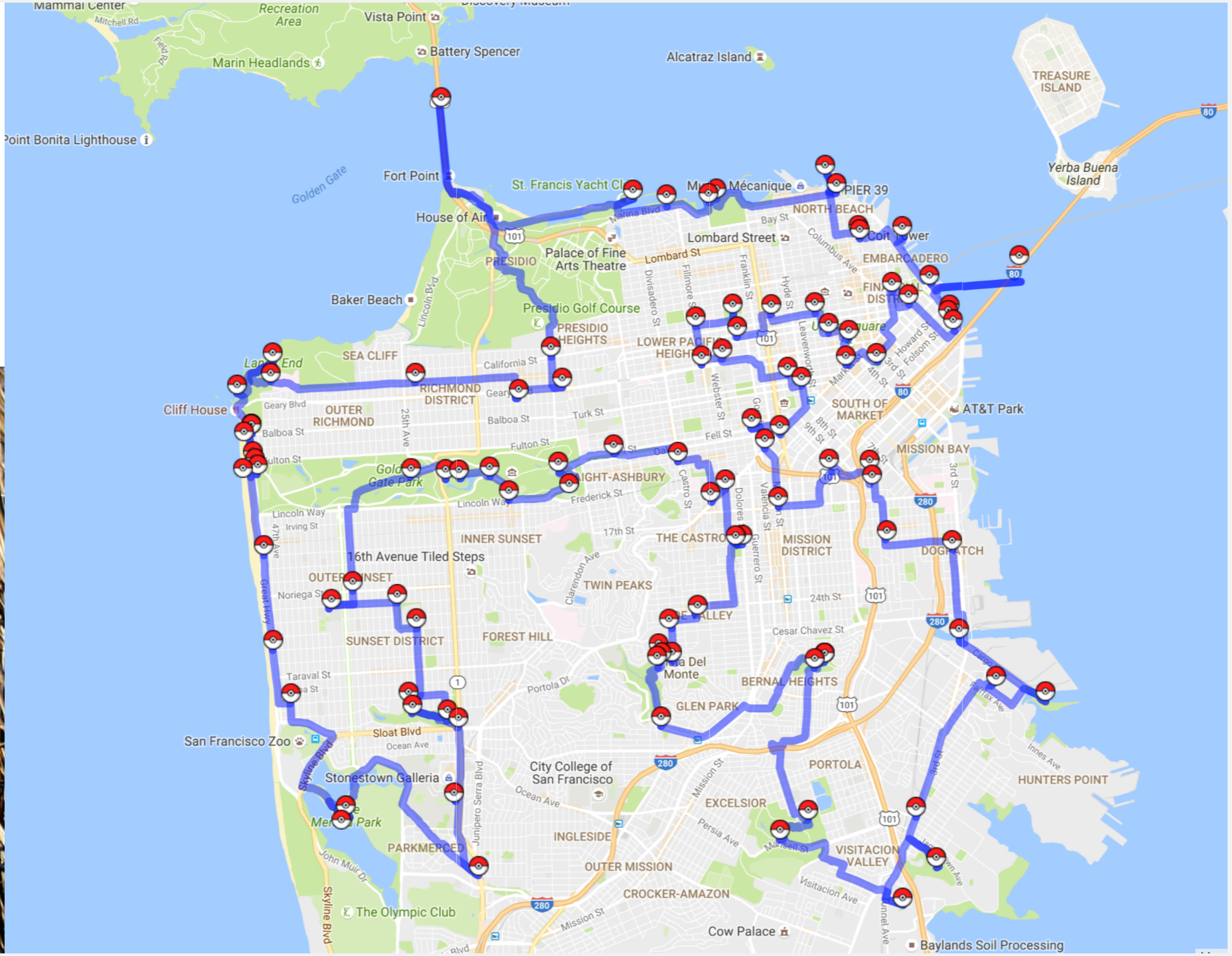
Assignment 8 Tips and Tricks

- *traveling salesperson problem*
- *nearest insertion heuristic*
- *smallest increase heuristic*
- *implementation*
- *beyond*

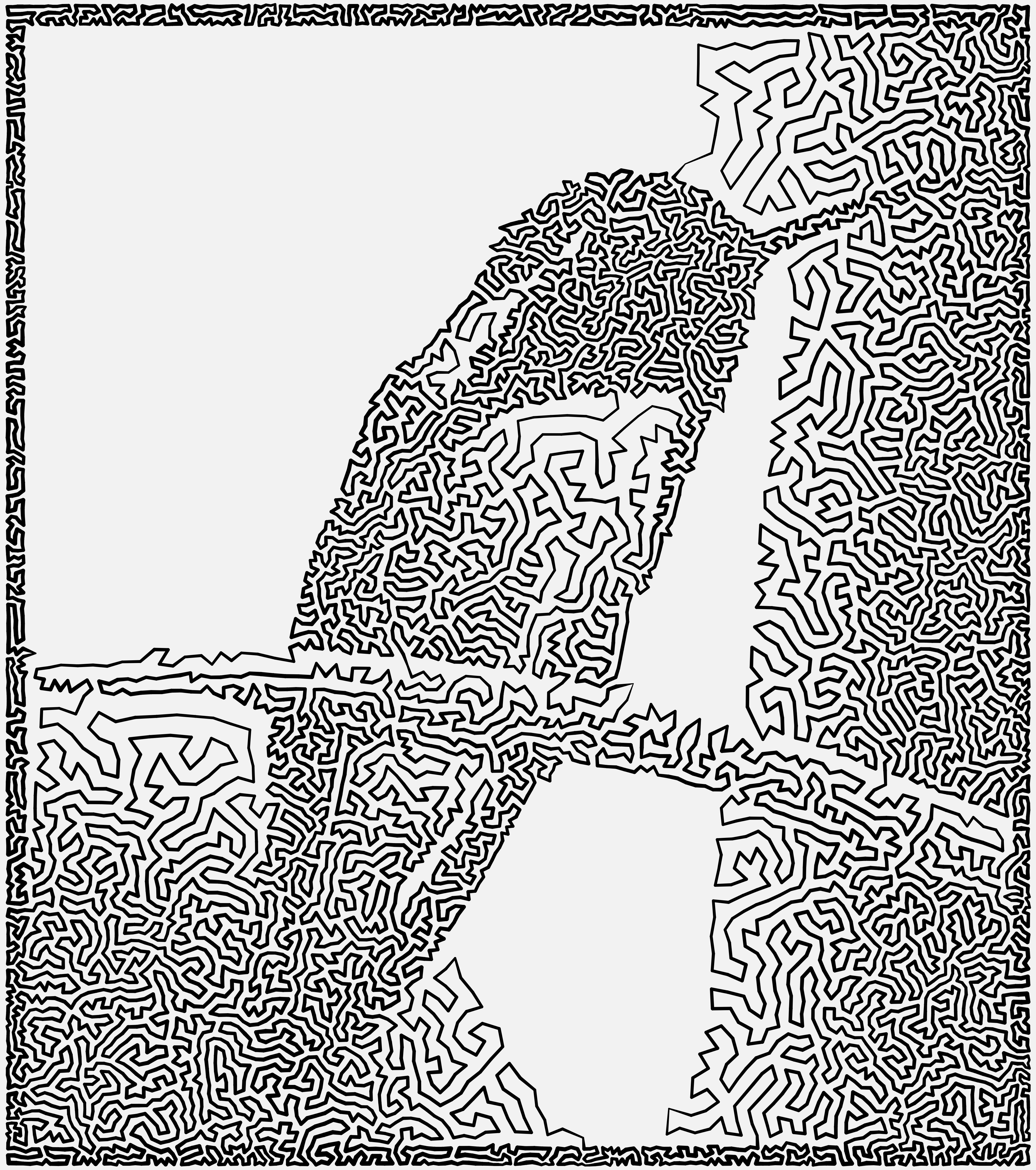
Pokemon Go

Map: Where to catch 123 Pokémon in San Francisco

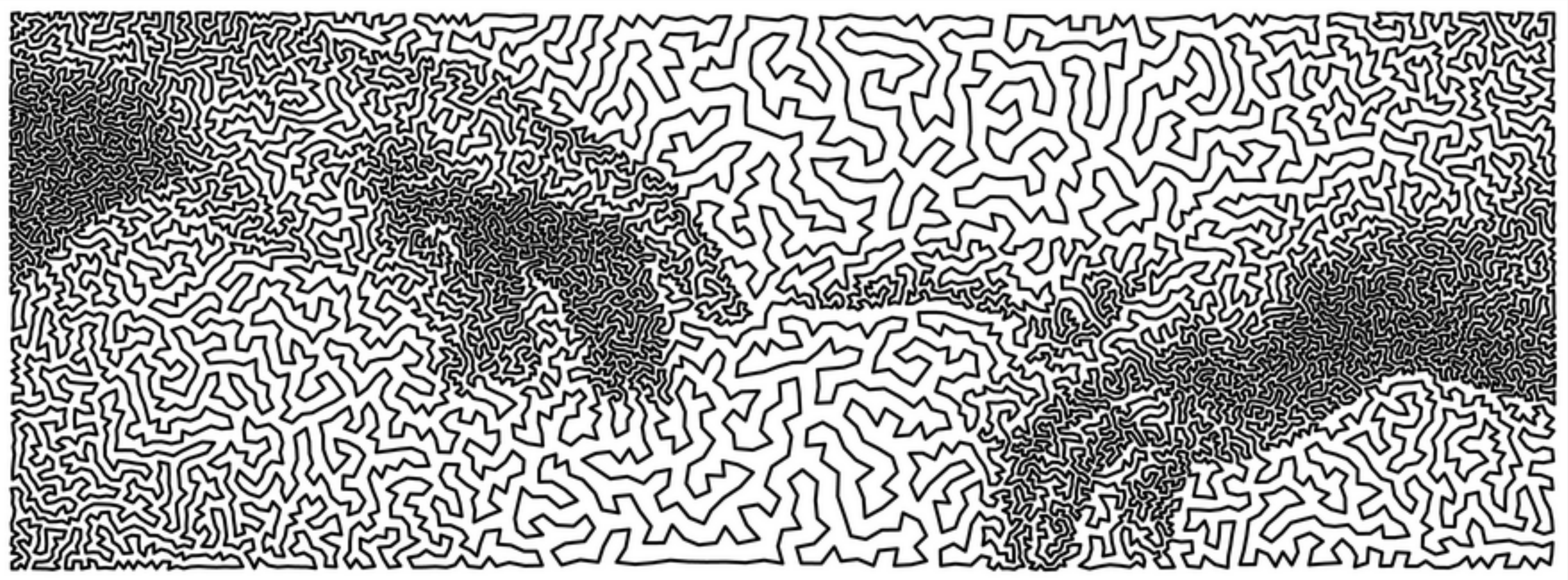
BY ADAM BRINKLOW | OCT 4, 2016, 6:33AM PDT



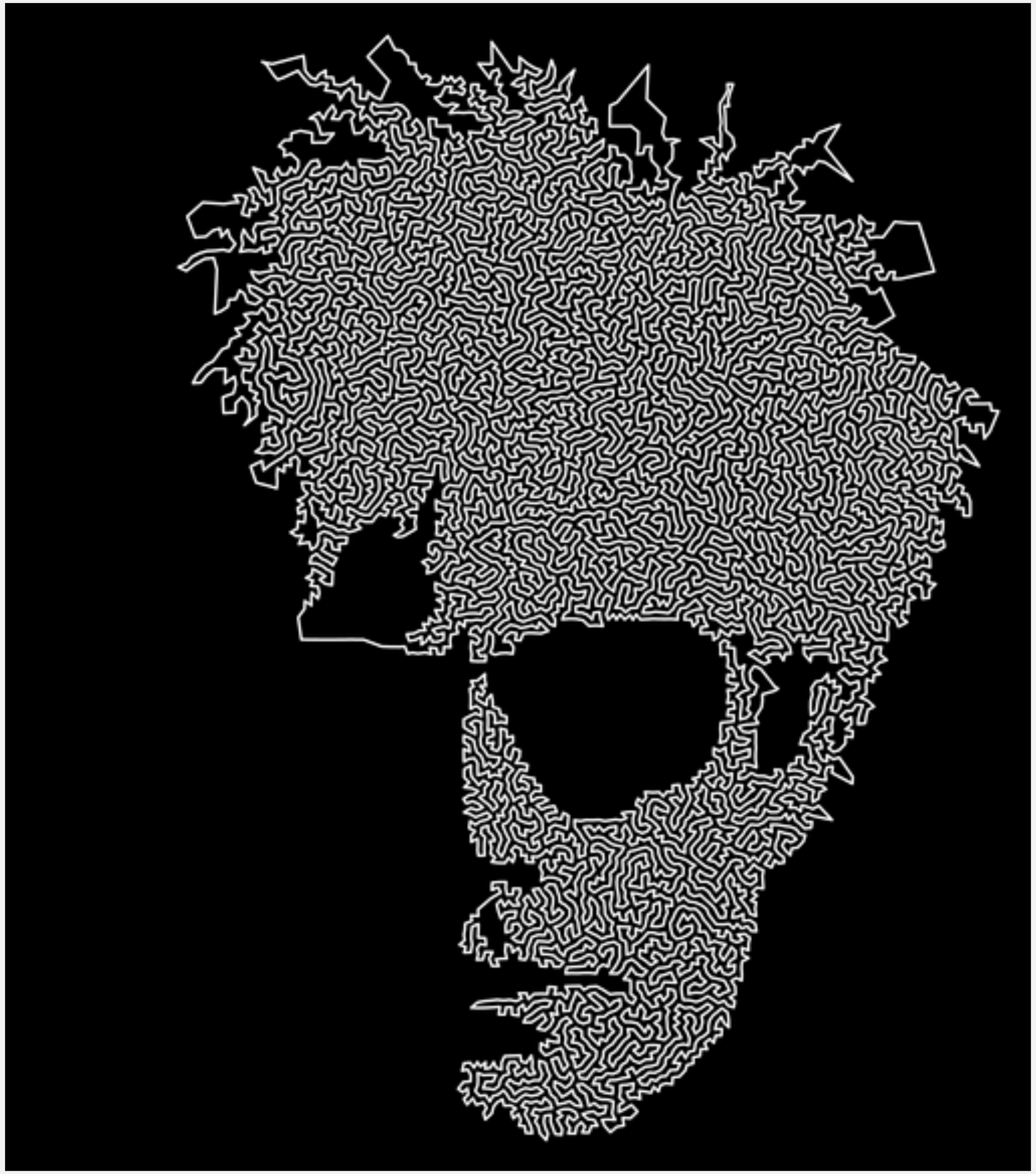
TSP art

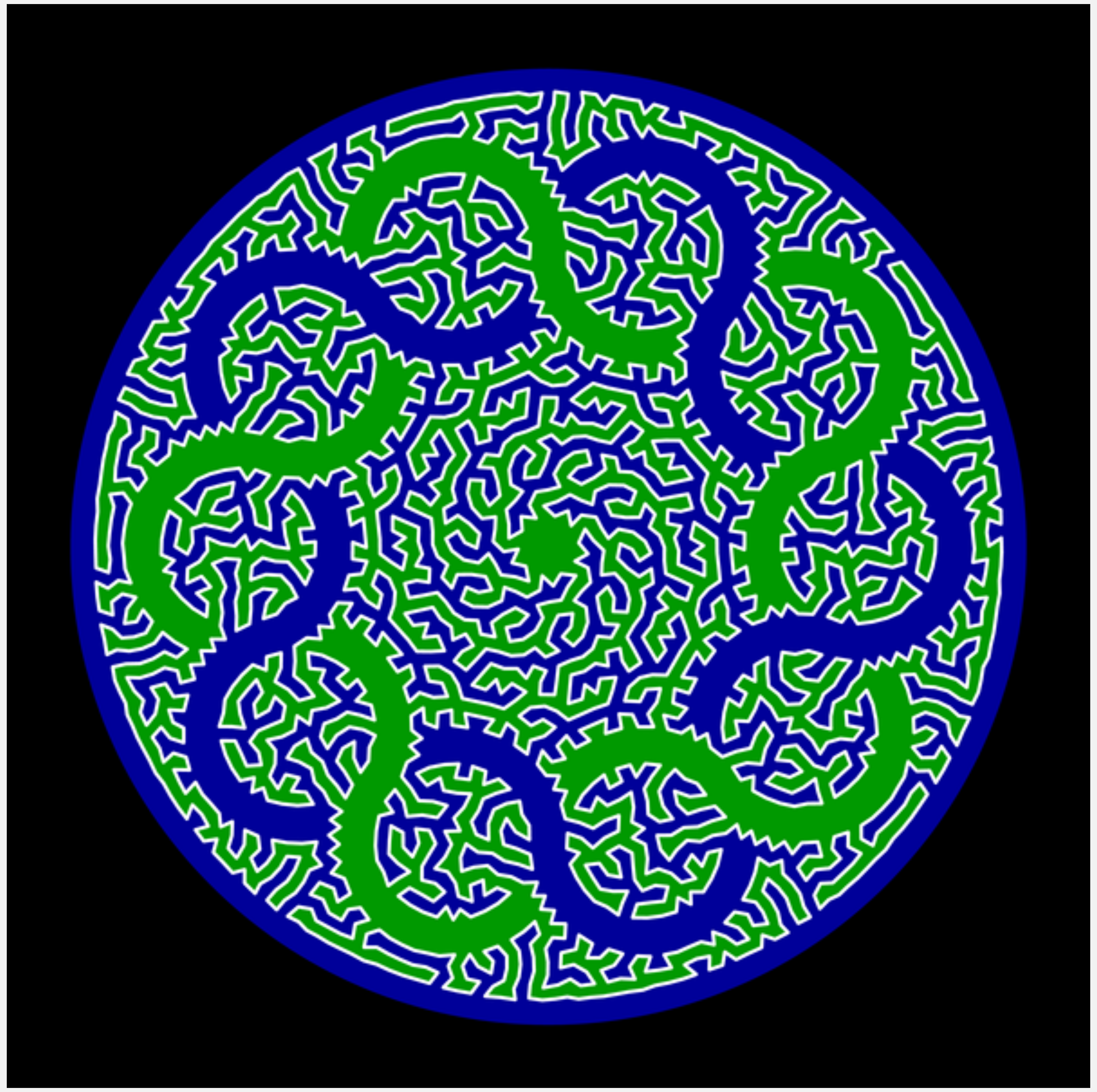


TSP art



TSP art





TSP books, apps, and movies