2.4 Compact storage of similar sequences

Consider the problem of storing a large set of strings, i.e., sequences of characters from a finite alphabet. We assume that the strings have many similar entries (they differ only in a small number of positions) and we wish to store them in a compact way. This problem arises in several contexts such as when storing DNA sequences, where the characters correspond to the four DNA bases. In this exercise, we consider the simplified version of the problem with only two characters.

Given a set of k sequences of M bits, we compute for each pair i, j, with $1 \leq i, j \leq k$, the Hamming distance between the sequences i and j, i.e., the number of bits that need to be flipped in sequence i to obtain sequence j. This function clearly satisfies the three usual properties of a distance: nonnegativity, symmetry and triangle inequality.

Consider the following set of 6 sequences and the corresponding matrix $D = \{d_{ij}\}$ of Hamming distances

1) `011100011101`
2) `101101011001`
3) `110100111001`
4) `101001111101`
5) `100100111101`
6) `010101011100`

	1	2	3	4	5	6
1	0	4	4	5	4	3
2		0	4	3	4	5
3			0	5	2	5
4				0	3	6
5					0	5
6						0

where, due to symmetry, only the upper triangle of the matrix is shown.

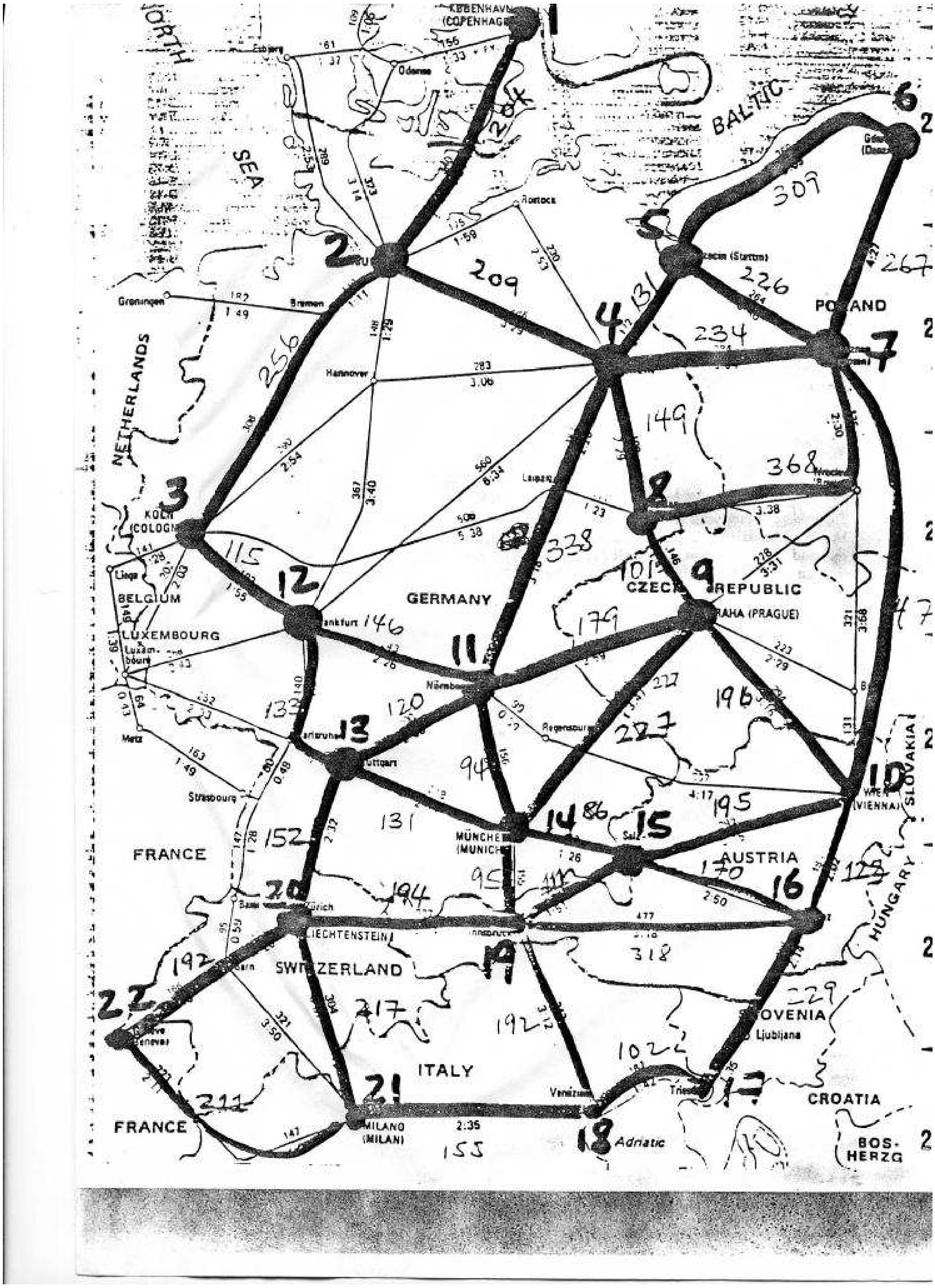
In order to exploit redundancies between sequences and to save memory, we can store: i) one of the sequences, called the reference sequence, completely and ii) for every other sequence, only the set of bit flips that allow us to retrive it either directly from the reference sequence or from another sequence.

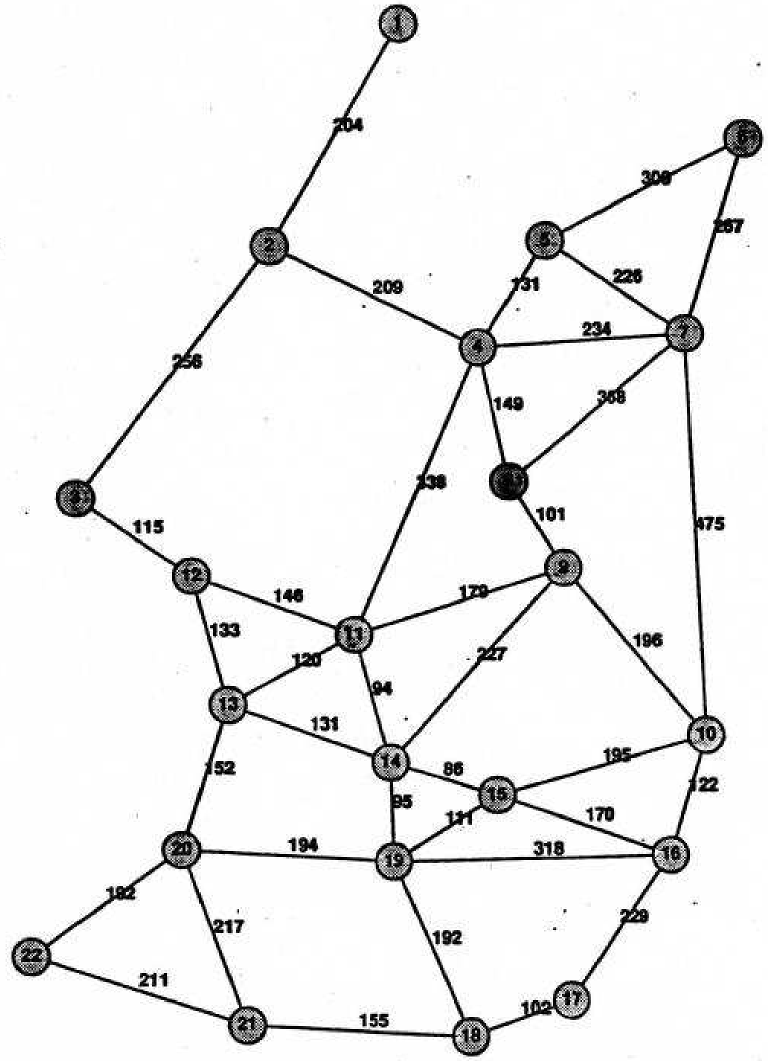
Show how the problem of choosing which differences to memorize, so as to minimize the total number of bits used for storage, can be reduced to the problem of finding a minimum-cost spanning tree in an appropriate graph. Solve the problem for the given instance.

[*Hint*: How many bits are needed to store the list of the differences between any given pair of sequences i and j?]

2.5 Minimum cost network design

Given the following undirected graph representing the possible fiber optic connections between some major European cities together with the corresponding distances. Since the fiber optic connection cost is proportional to the length of the connection, design a network of minimum total cost which guarantees that every pair of cities is connected.


KØBENHAVN (COPENHAGEN)
1
204
NORTH SEA
BALTIC
Esbjerg
Odense
Rostock
2
209
Szczecin (Stettin)
5
131
309
6
Gdansk
226
267
POLAND
4
234
7
Groningen
Bremen
256
Hannover
283
3.06
NETHERLANDS
149
368
Wroclaw
Leipzig
8
3
KÖLN (COLOGNE)
115
12
Liège
BELGIUM
LUXEMBOURG
Luxembourg
Frankfurt
146
GERMANY
328
101
CZECH REPUBLIC
9
PRAHA (PRAGUE)
179
11
Nürnberg
133
13
Karlsruhe
120
Stuttgart
94
227
196
Regensburg
Metz
Strasbourg
10
WIEN (VIENNA)
SLOVAKIA
14
86
15
195
MÜNCHEN (MUNICH)
Salzburg
131
152
FRANCE
AUSTRIA
170
16
128
HUNGARY
20
Basel
Zürich
194
95
Innsbruck
477
318
LIECHTENSTEIN
19
192
Bern
SWITZERLAND
217
22
Genève
192
229
SLOVENIA
Ljubljana
102
17
Trieste
CROATIA
21
ITALY
211
FRANCE
MILANO (MILAN)
2:35
155
Venezia
18
Adriatic
BOS. HERZG

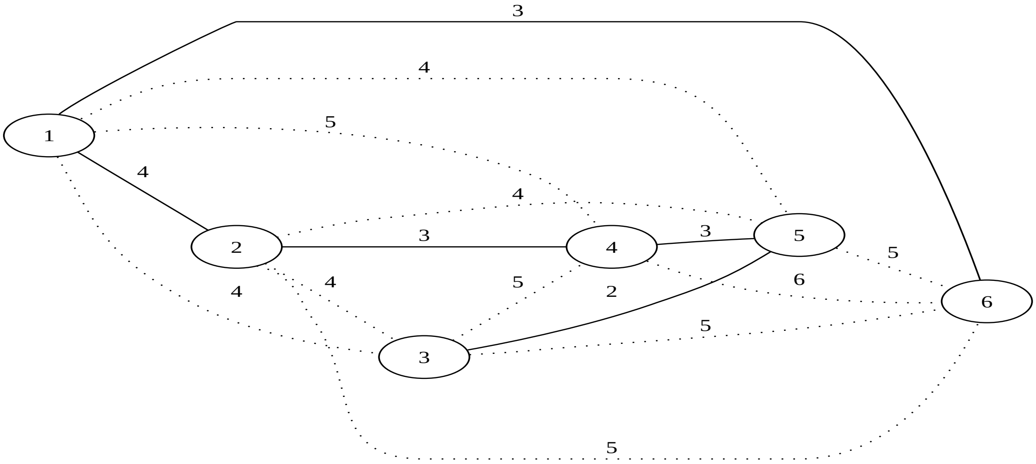

1
204
2
209
256
5
309
267
131
226
4
234
7
149
358
338
101
475
115
12
146
133
9
179
11
196
120
227
94
13
131
10
14
86
195
152
95
15
122
111
170
20
194
19
318
16
182
217
229
192
22
211
17
102
21
155
18

SOLUTION

2.4 **Compact storage of similar sequences**. We construct a complete graph G with a node for each sequence and an edge for each pair of sequences. Moreover, each edge (i,j) is assigned a cost d_{ij}.

The problem is then to look for a subgraph G' of G of minimum total cost. Since only one sequence is completely stored, G' must be connected to be able to construct any sequence. Since a subgraph of minimal cost is sought, G' will be acyclic. It followsa minimum-cost spanning tree in G provides an optimal solution to the problem under consideration.

The following minimum cost spanning tree has been found using Prim's algorithm.



2.5 **Minimum cost network design**. Apply either Prim's or Kruskal's algorithm to the given instance.