```
In []: !pip install biopython
!pip install matplotlib

In [18]: from Bio import Entrez, SeqIO
    from Bio.Align import PairwiseAligner
    from Bio import pairwise2
    import numpy as np
    import pandas as pd
    import random
    import matplotlib.pyplot as plt
```

1 Dalis

Iš paieškos sąrašo: https://www.ncbi.nlm.nih.gov/nuccore?term=33175%5BBioProject%5D+OR+33317%5BBioProject%5D sekų prieigos ID yra išsaugotos 'sequence.seq' faile. Iš failo pasirenkama dvidešimt 16S sekų ID.

```
In [50]: NUMBER_OF_SEQUENCES = 20

sequence_ids = []
with open('sequence.seq') as f:
    sequence_ids = f.readlines()
    sequence_ids = sequence_ids[:NUMBER_OF_SEQUENCES]
```

Pagal ID sekos yra gaunamos fasta formatu.

```
In []: sequences = []
    for seq_id in sequence_ids:
        handle = Entrez.efetch(db="nucleotide", id=seq_id, rettype="fasta", retmode="text")
        record = SeqIO.read(handle, "fasta")
        sequences.append(record)
```

Gaunami globalūs poriniai išlyginimai.

```
In [52]: aligner = PairwiseAligner()
    aligner.mode = 'global'
    alignment_table = np.empty((20, 20),dtype='int')
    for i in range(NUMBER_OF_SEQUENCES):
        for j in range(i, NUMBER_OF_SEQUENCES):
```

```
alignment_score = aligner.score(sequences[i], sequences[j])
alignment_table[i][j] = alignment_score
alignment_table[j][i] = alignment_score
```

Sukuriama lentelė su sekų eilės numeriais ir atitinkamomis išlyginimo kainomis.

```
In [53]: row_col_names = ["Seka " + str(i + 1) for i in range(NUMBER_OF_SEQUENCES)]
    df = pd.DataFrame(alignment_table,index = row_col_names, columns=row_col_names)
```

Atspausdinama lentelė

```
In [54]: print(df)
```

	Seka 1	Seka 2	Seka 3		Seka 5	Seka 6	Seka 7	Seka 8	\
Seka 1	1414	1159	1115	1125	1106	893	1148	1146	
Seka 2	1159	1475	1228	1243	1209	963	1324	1360	
Seka 3	1115	1228	1355	1222	1238	970	1233	1219	
Seka 4	1125	1243	1222	1344	1225	987	1261	1240	
Seka 5	1106	1209	1238	1225	1342	981	1238	1204	
Seka 6	893	963	970	987	981	1045	1045	970	
Seka 7	1148	1324	1233	1261	1238	1045	1466	1349	
Seka 8	1146	1360	1219	1240	1204	970	1349	1485	
Seka 9	1145	1359	1218	1239	1203	970	1349	1482	
Seka 10	904	1037	978	960	969	944	1030	1049	
Seka 11	907	1038	981	962	967	946	1033	1070	
Seka 12	899	1031	973	953	958	937	1025	1061	
Seka 13	1159	1348	1248	1267	1232	982	1364	1351	
Seka 14	1141	1316	1230	1235	1203	964	1284	1293	
Seka 15	1086	1171	1161	1168	1145	928	1165	1164	
Seka 16	1162	1364	1229	1249	1215	968	1347	1361	
Seka 17	1161	1361	1227	1247	1214	968	1345	1357	
Seka 18	1167	1346	1252	1319	1264	994	1380	1352	
Seka 19	1147	1316	1259	1263	1313	987	1353	1327	
Seka 20	1142	1314	1256	1265	1314	986	1350	1323	
	Seka 9	Seka 10	Seka 1					eka 15 '	١
Seka 1	1145	904	907	7 899	9 1	159	1141	1086	١
Seka 2	1145 1359	904 1037	907 1038	7 899 3 1033	9 1 1 1	159 348	1141 1316	1086 1171	١
Seka 2 Seka 3	1145 1359 1218	904 1037 978	907 1038 981	7 899 3 1033 L 973	9 1 1 1 3 1	159 348 248	1141 1316 1230	1086 1171 1161	\
Seka 2 Seka 3 Seka 4	1145 1359 1218 1239	904 1037 978 960	907 1038 983 962	7 899 3 1033 L 973 2 953	9 1 1 1 3 1 3 1	159 348 248 267	1141 1316 1230 1235	1086 1171 1161 1168	\
Seka 2 Seka 3 Seka 4 Seka 5	1145 1359 1218 1239 1203	904 1037 978 960 969	907 1038 983 962 967	7 899 3 1033 1 973 2 953 7 958	9 1 1 1 3 1 3 1 8 1	159 348 248 267 232	1141 1316 1230 1235 1203	1086 1171 1161 1168 1145	\
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6	1145 1359 1218 1239 1203 970	904 1037 978 960 969 944	907 1038 983 967 967	7 899 3 1033 L 973 2 953 7 958 5 933	9 1 1 1 3 1 3 1 8 1	159 348 248 267 232 982	1141 1316 1230 1235 1203 964	1086 1171 1161 1168 1145 928	\
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7	1145 1359 1218 1239 1203 970 1349	904 1037 978 960 969 944 1030	907 1038 983 967 967 946 1033	7 899 3 1033 L 973 2 953 7 958 5 933 3 1029	9 1 1 1 3 1 3 1 8 1 7 5 1	159 348 248 267 232 982 364	1141 1316 1230 1235 1203 964 1284	1086 1171 1161 1168 1145 928 1165	\
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8	1145 1359 1218 1239 1203 970 1349 1482	904 1037 978 960 969 944 1030 1049	903 1038 983 962 963 946 1033	7 899 8 1033 L 973 2 953 7 958 6 933 8 1029 9 1063	9 1 1 1 3 1 3 1 8 1 7 1 1 1	159 348 248 267 232 982 364 351	1141 1316 1230 1235 1203 964 1284 1293	1086 1171 1161 1168 1145 928 1165 1164	`
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9	1145 1359 1218 1239 1203 970 1349 1482 1484	904 1037 978 960 969 944 1030 1049	903 1038 983 963 963 1033 1076	7 899 3 1033 L 973 2 953 7 958 6 93 3 1029 9 1066	9 1 1 1 3 1 3 1 8 1 7 7 5 1 1 1	159 348 248 267 232 982 364 351 350	1141 1316 1230 1235 1203 964 1284 1293 1292	1086 1171 1161 1168 1145 928 1165 1164 1164	`
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10	1145 1359 1218 1239 1203 970 1349 1482 1484	904 1037 978 960 969 944 1030 1049 1048 1122	903 1038 983 963 946 1033 1076 1069	7 899 3 1033 L 973 2 953 7 958 5 933 3 1029 6 1066 L 1042	9 1 1 1 3 1 3 1 8 1 7 5 1 1 1 9 1	159 348 248 267 232 982 364 351 350	1141 1316 1230 1235 1203 964 1284 1293 1292 993	1086 1171 1161 1168 1145 928 1165 1164 1164	\
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11	1145 1359 1218 1239 1203 970 1349 1482 1484 1048	904 1037 978 960 969 944 1030 1049 1048 1122 1051	903 1038 963 963 944 1033 1076 1065 1126	7 899 3 1033 L 973 2 953 7 958 5 933 8 1029 9 1066 L 1044	9 1 1 1 3 1 3 1 8 1 7 7 5 1 1 1 2 1	159 348 248 267 232 982 364 351 350 037 045	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917	1
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11 Seka 12	1145 1359 1218 1239 1203 970 1349 1482 1484 1048 1069	904 1037 978 960 969 944 1030 1049 1048 1122 1051 1042	903 983 963 964 1033 1076 1069 1052	7 899 8 1033 L 973 2 953 7 958 5 933 8 1029 9 1066 L 1042 9 1116	9 1 1 1 3 1 3 1 7 7 5 1 1 1 2 1 9 1	159 348 248 267 232 982 364 351 350 037 045	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985 978	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917 907	1
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11 Seka 12 Seka 13	1145 1359 1218 1239 1203 970 1349 1482 1484 1048 1069 1060	904 1037 978 960 969 944 1030 1049 1048 1122 1051 1042 1037	903 983 963 963 944 1033 1076 1065 1126 1109	7 899 8 1033 1 973 2 953 7 958 6 933 8 1029 8 1066 1 1042 9 1116	9 1 1 1 3 1 3 1 7 5 1 1 1 2 1 9 1	159 348 248 267 232 982 364 351 350 037 045 040 539	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985 978 1306	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917 907 1173	
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11 Seka 12 Seka 13 Seka 14	1145 1359 1218 1239 1203 970 1349 1482 1484 1048 1069 1060 1350 1292	904 1037 978 960 969 944 1030 1049 1048 1122 1051 1042 1037 993	903 983 963 963 944 1033 1076 1065 1126 1109 1044	7 899 8 1033 1 973 2 953 7 958 6 933 8 1029 8 1066 1 1042 9 1116 9 1116 6 978	9 1 1 1 3 1 3 1 7 5 1 1 1 2 1 9 1 9 1	159 348 248 267 232 982 364 351 350 037 045 040 539 306	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985 978 1306 1442	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917 907 1173 1184	
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11 Seka 12 Seka 13 Seka 14 Seka 15	1145 1359 1218 1239 1203 970 1349 1482 1484 1048 1069 1060 1350 1292 1164	904 1037 978 960 969 944 1030 1049 1048 1122 1051 1042 1037 993 919	90: 1038 98: 96: 94: 103: 107: 106: 112: 110: 104: 98: 91:	7 899 8 1033 1 973 2 953 7 958 6 933 8 1029 8 1066 1 1046 9 1109 9 1110 5 978 7 907	9 1 1 1 3 1 8 1 7 5 1 1 1 2 1 9 1 9 1 0 1	159 348 248 267 232 982 364 351 350 037 045 040 539 306 173	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985 978 1306 1442 1184	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917 907 1173 1184 1297	
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11 Seka 12 Seka 13 Seka 14 Seka 15 Seka 16	1145 1359 1218 1239 1203 970 1349 1482 1484 1048 1069 1060 1350 1292 1164 1360	904 1037 978 960 969 944 1030 1049 1048 1122 1051 1042 1037 993 919	90: 1038 98: 96: 94: 103: 107: 106: 112: 110: 104: 98: 91:	7 899 8 1033 1 973 2 953 7 958 6 933 8 1029 8 1066 1 1046 9 1116 6 1046 7 907 7 1046	9 1 1 1 3 1 3 1 8 1 7 1 1 1 9 1 9 1 9 1 8 1 7 1	159 348 248 267 232 982 364 351 350 037 045 040 539 306 173 353	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985 978 1306 1442 1184 1311	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917 907 1173 1184 1297 1180	
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11 Seka 12 Seka 13 Seka 14 Seka 15 Seka 16 Seka 17	1145 1359 1218 1239 1203 970 1349 1482 1484 1069 1060 1350 1292 1164 1360 1356	904 1037 978 960 969 944 1030 1049 1048 1122 1051 1042 1037 993 919 1045 1045	903 983 963 964 1033 1076 1065 1126 1109 983 913 1044 1044	7 899 8 1033 1 973 2 953 7 958 5 933 8 1029 6 1066 1 1046 9 1116 5 1046 5 978 7 903 7 1046 6 103	9 1 1 1 3 1 3 1 7 7 5 1 1 1 9 1 9 1 9 1 7 1	159 348 248 267 232 982 364 351 350 037 045 040 539 306 173 353 351	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985 978 1306 1442 1184 1311 1309	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917 907 1173 1184 1297 1180 1179	
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11 Seka 12 Seka 13 Seka 14 Seka 15 Seka 16 Seka 17 Seka 18	1145 1359 1218 1239 1203 970 1349 1482 1484 1069 1060 1350 1292 1164 1360 1356 1351	904 1037 978 960 969 944 1030 1049 1048 1122 1051 1042 1037 993 919 1045 1045 1034	903 983 963 964 1033 1076 1069 1126 1109 1044 989 913 1044 1046	7 899 8 1033 L 973 2 953 7 958 6 933 8 1029 6 1066 L 1046 6 1109 7 1046 7 907 7 1046 1 1034	9 1 1 1 3 1 3 1 7 5 1 1 1 9 1 9 1 8 1 7 1 8 1	159 348 248 267 232 982 364 351 350 045 045 040 539 306 173 353 351 425	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985 978 1306 1442 1184 1311 1309 1305	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917 907 1173 1184 1297 1180 1179 1165	
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11 Seka 12 Seka 13 Seka 14 Seka 15 Seka 16 Seka 17 Seka 18 Seka 19	1145 1359 1218 1239 1203 970 1349 1482 1484 1069 1060 1350 1292 1164 1360 1356 1351 1326	904 1037 978 960 969 944 1030 1049 1048 1122 1051 1042 1037 993 919 1045 1045 1034 1024	90: 1038 98: 96: 94: 103: 107: 106: 110: 104: 98: 91: 104: 104: 104:	7 899 8 1033 1 973 2 953 7 958 6 933 8 1029 8 1066 1 1046 9 1106 6 1106 7 907 7 1046 6 1034 7 1016	9 1 1 1 3 1 3 1 8 1 7 1 1 2 1 9 1 9 1 8 1 7 1 9 1	159 348 248 267 232 982 364 351 350 037 045 040 539 306 173 353 351 425 373	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985 978 1306 1442 1184 1311 1309 1305 1281	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917 907 1173 1184 1297 1180 1179 1165 1160	
Seka 2 Seka 3 Seka 4 Seka 5 Seka 6 Seka 7 Seka 8 Seka 9 Seka 10 Seka 11 Seka 12 Seka 13 Seka 14 Seka 15 Seka 16 Seka 17 Seka 18	1145 1359 1218 1239 1203 970 1349 1482 1484 1069 1060 1350 1292 1164 1360 1356 1351	904 1037 978 960 969 944 1030 1049 1048 1122 1051 1042 1037 993 919 1045 1045 1034	903 983 963 964 1033 1076 1069 1126 1109 1044 989 913 1044 1046	7 899 8 1033 1 973 2 953 7 958 6 933 8 1029 8 1066 1 1046 9 1106 1 1046 7 903 7 1046 5 1039 1 1034 7 1016	9 1 1 1 3 1 3 1 8 1 7 1 1 2 1 9 1 9 1 8 1 7 1 9 1	159 348 248 267 232 982 364 351 350 045 045 040 539 306 173 353 351 425	1141 1316 1230 1235 1203 964 1284 1293 1292 993 985 978 1306 1442 1184 1311 1309 1305	1086 1171 1161 1168 1145 928 1165 1164 1164 919 917 907 1173 1184 1297 1180 1179 1165	

Seka 16 Seka 17 Seka 18 Seka 19 Seka 20

Seka	1	1162	1161	1167	1147	1142
Seka	2	1364	1361	1346	1316	1314
Seka	3	1229	1227	1252	1259	1256
Seka	4	1249	1247	1319	1263	1265
Seka	5	1215	1214	1264	1313	1314
Seka	6	968	968	994	987	986
Seka	7	1347	1345	1380	1353	1350
Seka	8	1361	1357	1352	1327	1323
Seka	9	1360	1356	1351	1326	1322
Seka	10	1045	1045	1034	1024	1025
Seka	11	1047	1046	1041	1017	1019
Seka	12	1040	1039	1034	1010	1011
Seka	13	1353	1351	1425	1373	1377
Seka	14	1311	1309	1305	1281	1276
Seka	15	1180	1179	1165	1160	1157
Seka	16	1496	1492	1362	1326	1325
Seka	17	1492	1494	1359	1323	1322
Seka	18	1362	1359	1500	1400	1402
Seka	19	1326	1323	1400	1500	1476
Seka	20	1325	1322	1402	1476	1500

2 Dalis

Užduočiai pasirinktas *Halopiger xanaduensis* archėjos genomas, kurio ID: NC_012654.1. Genomo seka atsisiunčiama.

```
In []: genome_id = "NC_012654.1"
    handle = Entrez.efetch(db="nucleotide", id=genome_id, rettype="fasta", retmode="text")
    genome = SeqIO.read(handle, "fasta")

In [57]: genome_lenght = len(genome.seq)
    T_lenght = 2000
    print(genome_lenght)
```

270022

Iš genomo atsitiktinai iškerpamas 2000 bp ilgio fragmentas **T**.

```
In [46]: cut_index = random.randint(0,genome_lenght - 2001)

T = genome[cut_index:cut_index + 2000]
genome_without_T = genome[:cut_index] + genome[cut_index + 2000:]
```

```
assert(len(genome) == len(T) + len(genome_without_T) and len(T) == 2000)

In [47]:
aligner = PairwiseAligner()
aligner.mode = 'local'
aligner.match_score = 1
aligner.mismatch_score = -1
aligner.gap_score = -2

Iš likusio genomo 10_000 kartų atsitiktinai parenkami 100 bp fragmentai, ir atliekami lokalūs išlyginimai su T.
```

```
In [48]: alignment_scores = np.empty((10000),dtype='int')
genome_without_T_lenght = len(genome_without_T)

for i in range(10000):
    fragment_index = random.randint(0,genome_without_T_lenght - 101)
    fragment = genome_without_T[fragment_index:fragment_index + 100]
    alignment_scores[i] = aligner.score(fragment,T)
```

Pavaizduojama išlyginimų kainų histograma

```
In [49]: plt.hist(alignment_scores, edgecolor='black')
    plt.xlabel('Alignment Score')
    plt.ylabel('Frequency')
    plt.title('Local Alignment Scores')
    plt.show()
```

