The Floyd-Warshall algorithm is well-known for computing shortest paths or transitive closure. For all-pairs shortest paths the computation looks like this:

```
// standard FW algorithm
function FW(C, N)
  for k=1:N
    for i=1:N
      for j=1:N
        C[i][j] = min(C[i][j], C[i][k]+C[k][j]);
```

For transitive closure (min, +) are replaced by (or, and). In fact, more useful variants can be obtained by changing the two operations. As long as some properties hold:

https://www.sciencedirect.com/science/article/pii/0304397577900561

The task is a fast implementation for at least the abovementioned two variants and the (Max, Min) version (check what it does). The (or, and) version should operate on bits, and ideally an AVX-512 version should be provided.

It looks like matrix-multiplication but the loop order cannot be exchanged. Blocking is still possible as explained (with other hints) here:

http://spiral.ece.cmu.edu:8080/pub-spiral/abstract.jsp?id=17

No need to build a full-fledged generator as there but some autotuning should be done.

Since the computation is rather simple, very careful optimization is expected.