6.5.2 Algorithm

Because each step in the path towards the goal costs the same (uses the same amount of energy/takes the same amount of time/etc), we can use a standard breadth-first search to find the shortest path. The first path found that starts at the source and ends at the goal will be the shortest path.[22] Starting at the source, the algorithm proceeds as follows:

1. Starting at the source, find all new cells that are reachable at distance 1, i.e. all paths that are just 1 unit in length, and mark them with that length.
2. Using the distance 1 cells, find all new cells which are reachable at distance 2.
3. Using all cells at distance 2 from the source, find all cells with distance 3.
4. Repeat until the target is found. This expansion creates a wavefront of paths that search broadly from the source cell until the target cell is hit.
5. From the target cell, select the shortest path back to the source and mark the cells along the path.

This 'wavefront' is called the fringe – the edge of what we've seen so far. At each iteration, we take a cell from the fringe and look at its undiscovered neighbours. Note that if it takes n steps to get to an item in the fringe, it then takes $n + 1$ steps to get to any of its undiscovered neighbours. By checking all paths of length n first, we can be sure that there is no quicker way to get to an undiscovered neighbour.[23] The fringe can be represented using a queue, this means that in iteration i, dequeue a cell from the fringe and enqueue all of its unvisited neighbours, which will have a path length of $i + 1$. Because the fringe is FIFO,