For the first cost function:

Cost = Math.pow(2.0, (getTile(p2) - getTile(p1)))

There will be totally three cases:
1. height1 > height2 : The cost will be really small but greater than 0.
2. height1 < height2 : The cost will be considerably big.
3. height1 = height2 : The cost will be 1.

For this cost function it will be more easy to go from a high point to a low point than go from a low point to a high point. So, the Heuristic function for this cost function will also be decided into two cases.

1. Current point is lower than final point:

In this case, the Base Heuristic function will be defined by adding the direct distance of the two points in a flat form and the cost function.

Base function: H = sqrt((x2 - x1)^2 + (y2 - y1)^2) + getcost(x, y)

However, This function is not the most heuristic one. Because the value for getcost(x, y) will be big for the height difference between x and y is big. However, we can achieve the height difference separately in every step in sqrt((x2 - x1)^2 + (y2 - y1)^2). Considering the cross can be made in only one step, so we just need to know the larger value of the point's coordinate. So the final function will be.

H = max((x2 - x1) , (y2 - y1)) * Math.pow(2, height(p2 - p1)/max((x2 - x1) , (y2 - y1)));

2. Current point is higher or at the same level with the final point.

Because the value for pow(2.0, (negative number) will be considerably small and the other two cases can be views equally, the function can be:

H = max((x2 - x1) , (y2 - y1))

These Heuristic functions I give is admissible because they will never overestimate the cost and always assume we have the most optimal choices after the step they are.

For the first cost function:

Cost = (double) (getTile(p1)) / ((double) getTile(p2)+1.0);

There will be totally three cases:
1. height1 > height2 :
2. height1 < height2 :
3. height1 = height2 :

in these states, the cost will be height1/(height2 + 1)

1. Current point is higher than final point:

Height1 will larger than height2 and the cost may mostly become larger than 1 or close to one. But there is an optimal case: these two height are close to one and height1 is slightly larger than height2. In this case, the outcome will become larger but not smaller than 1/2.

2. Current point is lower than final point:

In this case, the cost height1/(height2 + 1), in this case the movement cost of tow points can be also reduced to larger or equal to 1/2.

3. Current point is at the same level with the final point

At this point, the cost will be height1/(height2 + 1). and the out come will largely close to 1. However, the most optimal solution for this part is both height1 and height2 are equal to 1. So the cost will be minimized to 1/2.

In conclusion, my Heuristic function will never overestimate the cost and will always assume the rest chose is the most optimal choice. The function for dividend will be:

H = max((x2 - x1) , (y2 - y1)) /2