* Original Heuristic:

Instead of simply calculating Manhattan distance (MD), which is the sum of the absolute differences between the two vectors, we can find a closer cost added to MD.

H’(n)=

Relax the problem by simplifying the 1\*2 pieces to two 1\*1 pieces, there are only one 2\*2 pieces, and all rest pieces are 1\*1 (labeled ‘x’).

* Say MD=0, we are at goal state, actual cost is 0.
* Say MD =1, the best case, we can move to goal state at cost of 1.
  + Ex. xxxx xxxx

xxxx xxxx

x11x ----down-🡪 x00x

x11x cost:1 x11x

x00x x11x

* Say MD=2, the best case, we can move to goal state in 5 moves
  + Ex1. xxxx xxxx xxxx xxxx

xxxx xxxx xxxx xxxx

110x. ---right -🡪 011x -move empty 🡪 x11x --down-> x00x

110x. cost: 1 011x cost: 3 x11x cost: 1 x11x

xxxx xxxx x00x x11x

* + Ex2. xxxx xxxx xxxx xxxx

x11x x00x xxxx xxxx

x11x. ---right -🡪 x11x -move empty 🡪 x11x --down-> x00x

x00x. cost: 1 x11x cost: 6 x11x cost: 1 x11x

xxxx xxxx x00x x11x

* In best case for MD>= 2, 2\*2 piece need **at least cost of 3** to swap the empty pieces with 1\*1 pieces and prepare the empty pieces for the next move.
* Why admissible?

H’(n) is admissible as it never overestimates the cheapest cost from state n to a goal state.

In the relaxed rules, for MD>=2, the 2\*2 piece needs to move at least **MD** moves, and between **(MD-1)** pairs of consecutive moves we need to move the empty pieces, with a **min cost of 3** each.

* Why does it dominate Manhattan distance heuristic?
  + For all possible number of MD numbers, H’(n) >= H(n); and for MD>=2, H’(n) >H(n).
* Implementation:
  + get\_original\_heuristic(board)
  + A\* with original heuristic for the classic HRD config still runs at costs of 116.