The key idea here is to convert each of these infinite 2-dimensional tapes into 1-dimensional tapes. Since the given Turing machine computes f in time $T(n)$, that means each tape's pointer doesn't move up more than $T(n)$ steps, doesn't move right more than $T(n)$ steps, doesn't move down more than $T(n)$ steps, and doesn't move left more than $T(n)$ steps, for a resulting upper-bounding, square-shaped space of $4T(n)^2 + 4T(n) + 1 = O(T(n)^2)$ cells around the starting point for the pointer to move around. In our new Turing machine, then, we start with the register pointing to the $(2T(n)^2 + 2T(n) + 1)$'th cells in each tape (the center of the bounding square). Whenever the original Turing machine moves up, our transition function makes it move left $2T(n) + 1$; whenever it moves down, we make it move right $2T(n) + 1$; whenever it moves left, we move left just the same; and whenever it moves right, we move right just the same. The original Turing machine moves up or down at most $T(n)$ times, and for each of those, our Turing machine does it in $O\big(T(n)\big)$ time, making our Turing machine compute f in $O\big(T(n) \cdot T(n)\big) = O(T(n)^2)$ time.