Lemma 1. *Let $s \neq t$, and $s \nrightarrow t$. Then, $s.v[s.p] > t.v[s.p]$*

Proof. The more interesting case is when s and t occur on two different processes ($s.p \neq t.p$).

$s \nrightarrow t$ means that there is no message path from s to t.
Process s.p's (say, Process-0) component

A local timestamp (s.v[s.p]) essentially counts local events. On receiving a message from s, t takes component wise max, and increments only its own local index.

Thus, for all other processes t: $s.v[s.p] \geq t.v[s.p]$.

Now, the only way for equality $s.v[s.p] = t.v[s.p]$, is if there is a message path from s to t. But we know that $s \nrightarrow t$ and thus no message path exists.

Thus, we get strict inequality: $s.v[s.p] > t.v[s.p]$ $\square$

Theorem 2. *$s \rightarrow t$ if and only if $s.v < t.v$*

Proof. Let's start with the forward direction:
If $s \rightarrow t$, then $s.v < t.v$

$s \rightarrow t$ means that there is some message path from s to t.
If we apply the rules of vector clock updates, we get:

$\forall k$: $s.v[k] \leq t.v[k]$, because again rcpt involves pairwise maximums.
We thus get that $s.v \leq t.v$. But we want strict inequality.

But, we can apply Lemma 1.
$s \rightarrow t$ means that $t \nrightarrow s$, which means: $t.v[t.p] > s.v[t.p]$
Thus there is one index for which strict inequality occurs, and thus we get $s.v < t.v$

Now the other direction: **If $s.v < t.v$, then $s \rightarrow t$**

We can prove this by contradiction and Lemma 1.
Suppose $s \nrightarrow t$. Then we get $s.v[s.p] > t.v[s.p]$
But this contradicts $s.v < t.v$

□