The number of ways to partition \(a+b\) things into \(a\) things and \(b\) things is \((a+b)! / (a!b!)\) where ! is the factorial function. First without time.

\[
x := (a+b)! / (a!b!) \iff
\]

\[
\text{if } a = 0 \text{ then } x := 1
\]

\[
\text{else } a := a - 1. \quad x := (a+b)! / (a!b!). \quad a := a + 1. \quad x := x(a+b)/a \quad \Box
\]

The assignment \(x := (a+b)! / (a!b!)\) means \(x' = (a+b)! / (a!b!)\) \& \(a' = a \& b' = b\). On the right side it is a recursive call. Stating it as an assignment makes the proof easy: just use the substitution law and simplify. The proof is by cases. First case:

\[
a = 0 \land (x := 1) \implies (x := (a+b)! / (a!b!)) \quad \text{definition of assignment}
\]

\[
\Box = a = 0 \land x' = 1 \land a' = a \land b' = b \implies x' = (a+b)! / (a!b!) \land a' = a \land b' = b \quad \text{use 0!=1}
\]

Now the time.

\[
t' = t + a \iff
\]

\[
\text{if } a = 0 \text{ then } x := 1
\]

\[
\text{else } a := a - 1. \quad t := t + 1. \quad t' = t + a. \quad a := a + 1. \quad x := x(a+b)/a \quad \Box
\]

Proof by cases. First case:

\[
a = 0 \land (x := 1) \implies t' = t + a \quad \text{definition of assignment}
\]

\[
\Box = a = 0 \land x' = 1 \land a' = a \land b' = b \land t' = t \implies t' = t + a
\]

\[
\Box = t
\]

Second case, starting with the right side:

\[
a = 0 \land (a := a - 1. \quad t := t + 1. \quad t' = t + a. \quad a := a + 1. \quad x := x(a+b)/a) \quad \text{assignment}
\]

\[
\Box = a = 0 \land (a := a - 1. \quad t := t + 1. \quad t' = t + a. \quad a := a + 1. \quad x := x(a+b)/a \land a' = a \land b' = b \land t' = t)
\]

\[
\Box = a = 0 \land (t' = t + a. \quad x' = x(a+1+b)/(a+1) \land a' = a + 1 \land b' = b \land t' = t) \quad \text{sequential composition}
\]

\[
\Box = a = 0 \land (\exists x'. \quad a' = a \land b' = b \land t' = t'. \quad t' = t + a \land x' = x(a+1+b)/(a+1) \land a' = a'+1 \land b' = b' \land t' = t') \quad \text{one point 4 times}
\]

\[
\Box = t' = t + a
\]

When refining \(x := (a+b)! / (a!b!)\), there was no time variable. Adding the time variable, we cannot write this as an assignment, because that would mean \(t' = t\). We can put the result and the timing together as

\[
x' = (a+b)! / (a!b!) \land a' = a \land b' = b \land t' = t+a
\]

or as

\[
x := (a+b)! / (a!b!). \quad t := t+a
\]

Here is a solution that is symmetric in \(a\) and \(b\).

\[
x := (a+b)! / (a!b!) \iff
\]

\[
\text{if } a = 0 \lor b = 0 \text{ then } x := 1
\]

\[
\text{else } a := a - 1. \quad b := b - 1. \quad x := (a+b)! / (a!b!).
\]

\[
a := a + 1. \quad b := b + 1. \quad x := x(a+b)/x(a+b+1)(a+b)
\]

And its execution time is smaller: \(a \downarrow b\).
Here is a solution with the same execution time and its recursion does not require a stack.
\[ x' = (a+b)! / (a!b!) \land t' = t + a↓b \iff 
\]
\[ x := 1. \quad x' = x \times (a+b)! / (a!b!) \land t' = t + a↓b \]
\[ x' = x \times (a+b)! / (a!b!) \land t' = t + a↓b \iff 
\]
\[ \text{if } a=0 \lor b=0 \text{ then ok } 
\text{else } x := x/a \times (a+b-1) \times (a+b). \quad a := a-1. \quad b := b-1. \quad t := t+1. \]
\[ x' = x \times (a+b)! / (a!b!) \land t' = t + a↓b \]

Now, here is a for-loop solution. Define invariant
\[ A_k = x = (a+k)! / (a!k)! \]
Then
\[ x' = (a+b)! / (a!b!) \iff x := 1. \quad A_0 \Rightarrow A' \]
\[ A_0 \Rightarrow A' \iff \text{for } k := 0;..b \text{ do } 0;..b \land A_k \Rightarrow A'(k+1) \text{ od } 
\]
\[ k : 0;..b \land A_k \Rightarrow A'(k+1) \iff x := x \times (a+k+1)/(k+1) \]
with timing \( t' = t+b \).

Finally, here are two functional solutions. Define
\[ f = \langle a, b : \text{nat} \cdot (a+b)! / (a!b)! \rangle \]
Then
\[ f a b = \text{if } a=0 \text{ then } 1 \text{ else } f(a-1) \times b \times (a+b) / a \]
with execution time \( a \). For execution time \( a↓b \)
\[ f a b = \text{if } a=0 \lor b=0 \text{ then } 1 \text{ else } f(a-1) \times (b-1) \times (a+b-1) \times (a+b) / a / b \]