Machine := tuple of *devices* $(d_1, \ldots, d_k)$

Device $\begin{cases} \text{Realm} := \text{the set of } \textit{states} \\ \text{Repetory} := \text{the set of } \textit{operations} \end{cases}$

Operation := a partial function on the realm

Program on machine $M = (d_1, \ldots, d_k)$ $\begin{cases} \text{Initializer } \alpha = (\alpha_1, \ldots, \alpha_k) \\ \quad \text{where } \alpha_i \text{ is a function from the possible arguments of } P \text{ into the realm of } d_i \\ \text{finite set } I \text{ of instructions } (f_1, \ldots, f_k) \\ \quad \text{where } f_i \text{ is an operator of device } d_i \text{ or the special operator NOOP, doing nothing} \\ \text{Terminator } \omega = (\omega_1, \ldots, \omega_k) \\ \quad \text{where } \omega_i \text{ is a relation between the realm of } d_i \text{ and the possible results of } P \end{cases}$

$$\text{argument} \longrightarrow \boxed{\text{program}} \longrightarrow \text{result}$$

The *argument* of a program P can be a graph, a number, etc., and likewise for the *result*.

The *initializer* tells how the argument is encoded as initial state for each of M's devices.

The *instructions* tell how P goes from state to state.

The *terminator* tells for each of M's devices which are its final states, and decodes a combination of final states into a *result* of P.

Configuration := tuple of states $(s_1, \ldots, s_k)$ of M's devices, representing global state

History := $(x, \alpha, C_0, \pi_1, C_1, \ldots, C_{n-1}, \pi_n, C_n, \omega, y)$ with $x\alpha = C_0$, $C_i\alpha_{i+1} = C_{i+1}$ and $C_n\omega = y$.

Transfer relation: $x \overset{\tau}{\mapsto} y$

Partial History := $(x, \alpha, C_0, \pi_1, C_1, \ldots, C_{n-1}, \pi_n, C_n)$

Infinite History := $(x, \alpha, C_0, \pi_1, C_1, \pi_2, C_2, \ldots)$