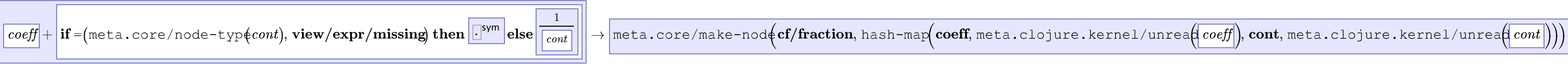Continued fraction constructor, with coefficient (an int at runtime) and optional continuation (a cont. fraction).

$$expr \leftarrow fraction\{coeff: expr, cont: expr\}$$

$$\boxed{\boxed{coeff} + \boxed{\textbf{if } = \big(\texttt{meta.core/node-type}(cont), \textbf{view/expr/missing}\big) \textbf{ then } \boxed{.}^{\textsf{sym}} \textbf{ else } \frac{1}{\boxed{cont}}}} \rightarrow \boxed{\texttt{meta.core/make-node}\big(\textbf{cf/fraction}, \texttt{hash-map}\big(\textbf{coeff}, \texttt{meta.clojure.kernel/unread}\big(\boxed{coeff}\big), \textbf{cont}, \texttt{meta.clojure.kernel/unread}\big(\boxed{cont}\big)\big)\big)}$$

$$expr \leftarrow coeff\{expr: expr\}$$

$$\boxed{\textbf{coeff}^{\textsf{kw}}\boxed{\boxed{expr}}^{\textsf{juxt}}} \rightarrow \boxed{\textbf{attr}\boxed{\boxed{expr}, \textbf{coeff}}}$$

$$expr \leftarrow cont\{expr': expr\}$$

$$\boxed{\textbf{cont}^{\textsf{kw}}\boxed{\boxed{expr'}}^{\textsf{juxt}}} \rightarrow \boxed{\textbf{attr}\boxed{\boxed{expr'}, \textbf{cont}}}$$

$$expr \leftarrow ratio\{expr'': expr\}$$

$$\boxed{\textbf{ratio}^{\textsf{kw}}\boxed{\boxed{expr''}}^{\textsf{juxt}}} \rightarrow \boxed{\textbf{let } r = \left(\textbf{fn } f \rightarrow \left(\textbf{if type}\big[f\big] = \textbf{cf/fraction then eval}\big[\textbf{coeff}[f]\big] + \frac{1}{q\big(\textbf{cont}\big[f\big]\big)} \textbf{ else eval}\big[f\big]\right)\right) \textbf{ in } r\left(\boxed{expr''}\right)}$$