Pattern Non–Redundancy

We now introduce a judgment expressing that *no branch of a pattern list is redundant*, i.e. each pattern contributes new, previously uncovered values.

We write a sequence of patterns as

$$\vec{p} = (p_1, \ldots, p_n).$$

Recall the (partial) matching function

$$\mathsf{match}(p, v) = \theta$$

which is defined precisely when p matches value v and returns a binding environment θ.

Non–Redundancy Judgment

We define a judgment

$$\Delta \vdash \mathsf{NR}(T, \vec{p})$$

read: *under type context Δ, the pattern sequence $\vec{p}$ is non–redundant for type T.*

Intuitively, each pattern p_k must match at least one value of type T that is not matched by any earlier pattern $p_1, \ldots, p_{k-1}$.

We define NR inductively.

Empty Sequence The empty sequence is trivially non–redundant:

$$[\text{NR-Empty}]\Delta \vdash \mathsf{NR}(T, ())$$

Extending a Non–Redundant Sequence Suppose we already have a non–redundant sequence $(p_1, \ldots, p_{k-1})$. We may extend it with p_k provided that there exists a value of type T which matches p_k but does *not* match any of the earlier patterns.

Formally:

$$[\text{NR-Cons}]\Delta \vdash \mathsf{NR}(T, (p_1, \ldots, p_{k-1}))\exists v : T.\ \exists \theta.\ \mathsf{match}(p_k, v) = \theta\ \wedge\ \forall i < k.\ \mathsf{match}(p_i, v) \text{ is undefined}\Delta \vdash$$

Thus, each new pattern is required to introduce at least one *fresh* value of type T that was not already covered.

Typing Rule with Exhaustiveness and Non–Redundancy

Combining the earlier exhaustiveness judgment Exh with non–redundancy, we may state a strengthened typing rule for pattern matching.

$$[\text{T-Match-Total}]\Gamma \vdash e : T\forall k.\ \Gamma, \Gamma_{p_k} \vdash e_k : U\Delta \vdash \mathsf{Exh}(T, \{p_1, \ldots, p_m\})\Delta \vdash \mathsf{NR}(T, (p_1, \ldots, p_m))\Gamma \vdash \texttt{match}$$

Here:

- Exh guarantees that *every* value of type T is matched by *some* pattern.
- NR guarantees that *each* branch matches *at least one* value that no earlier branch matches.

Non–Redundancy Lemma

We may now formulate a meta–theoretic property of the non–redundancy judgment.

Lemma (Branch Non–Redundancy). Let T be a type and $\vec{p} = (p_1, \ldots, p_m)$ a sequence of patterns such that

$$\Delta \vdash \mathsf{NR}(T, \vec{p}).$$

Then for each k with $1 \leq k \leq m$ there exists a value $v_k : T$ and a binding environment θ_k such that:

$$\mathsf{match}(p_k, v_k) = \theta_k \quad \text{and} \quad \forall i < k.\ \mathsf{match}(p_i, v_k) \text{ is undefined.}$$

In other words, every branch $p_k \Rightarrow e_k$ is *semantically reachable*: there is at least one value of type T for which operational evaluation of the corresponding `match` expression selects that branch.

Sketch of Proof. The proof proceeds by induction on the derivation of $\Delta \vdash \mathsf{NR}(T, \vec{p})$.

- In case NR-Empty, the statement is vacuously true.
- In case NR-Cons, the induction hypothesis yields the required witnesses v_i for all $i < k$, and the side condition of the rule provides the witness v_k for the new pattern.

Thus, $\mathsf{NR}(T, \vec{p})$ precisely captures the intended notion of branch non–redundancy in pattern matching.