Initial Type Assumption A_0

$$
\begin{aligned}
A_0(x) &= \forall a.\, a \quad \text{for all } x \in \mathcal{V} \\
A_0(c) &= \text{pre-defined type schema in } \textsc{Haskell}\text{, for all } c \in \mathcal{C}_0 \\
A_0(\underline{\mathsf{constr}}) &= \forall\, (\underline{\mathsf{type}}_1 \to \ldots \to \underline{\mathsf{type}}_n \to (\underline{\mathsf{tyconstr}}\, a_1 \ldots a_m)), \\
A_0(\mathtt{bot}) &= \forall a.\, a \\
A_0(\mathtt{isa}_{(\,)}) &= (\,) \to \mathtt{Bool} \\
A_0(\mathtt{if}) &= \forall a.\, \mathtt{Bool} \to a \to a \to a \\
A_0(\mathtt{fix}) &= \forall a.\, (a \to a) \to a \\
A_0(\mathtt{isa}_{\underline{\mathsf{constr}}}) &= \forall\, ((\underline{\mathsf{tyconstr}}\, a_1 \ldots a_m)\ \texttt{->}\ \mathtt{Bool}) \\
A_0(\mathtt{argof}_{\underline{\mathsf{constr}}}) &= \forall\, ((\underline{\mathsf{tyconstr}}\, a_1 \ldots a_m)\ \texttt{->}\ (\underline{\mathsf{type}}_1, \ldots, \underline{\mathsf{type}}_n)) \\
A_0(\mathtt{sel}_{n,i}) &= \forall a_1 \ldots a_n.\, (a_1, \ldots, a_n) \to a_i \\
A_0(\mathtt{tuple}_n) &= \forall a_1 \ldots a_n.\, a_1 \to \ldots \to a_n \to (a_1, \ldots, a_n)
\end{aligned}
$$

Here, $\underline{\mathsf{constr}}$ is introduced by the declaration

$$\mathtt{data}\ \underline{\mathsf{tyconstr}}\, a_1 \ldots a_m = \ \ldots \mid \underline{\mathsf{constr}}\ \underline{\mathsf{type}}_1 \ldots \underline{\mathsf{type}}_n \mid \ldots$$

Type Inference Algorithm $\mathcal{W}$

Let A be a type assumption, let $t \in \Lambda$.
$\mathcal{W}(A, t)$ is either a pair (θ, τ) or the computation fails because of a failing unification problem. Let $c \in \mathcal{C} \cup \mathcal{V}$.

- $\mathcal{W}(\, A + \{c :: \forall a_1, \ldots, a_n . \tau\},\ c\,) \ = \ (\, id,\ \tau[a_1/b_1, \ldots, a_n/b_n]\,)$,
  $b_1, \ldots, b_n$ are new variables

- $\mathcal{W}(\, A,\ \lambda x.\, t\,) \ = \ (\, \theta,\ b\theta \rightarrow \tau\,)$,
  where $\mathcal{W}(\, A + \{x :: b\},\ t\,) = (\theta, \tau)$, b is a new variable

- $\mathcal{W}(\, A,\ (t_1\, t_2)\,) \ = \ (\, \theta_1\, \theta_2\, \theta_3,\ b\theta_3\,)$,

  where
  $$\begin{aligned} \mathcal{W}(\, A,\ t_1\,) &= (\, \theta_1,\ \tau_1\,) \\ \mathcal{W}(\, A\,\theta_1,\ t_2\,) &= (\, \theta_2,\ \tau_2\,) \\ \theta_3 &= mgu(\tau_1\, \theta_2,\ \tau_2 \rightarrow b), \\ & \ b \text{ is a new variable.} \end{aligned}$$

Example

$$
\begin{array}{rcl}
\mathcal{W}(A_0, \texttt{fix}\,(\lambda\,\texttt{fact}\,x.\ \texttt{if}\ (x\ \texttt{<=0})\ \texttt{1}\ (\texttt{fact}\,(x - \texttt{1}) * x))) & = & ([\ldots], \texttt{Int} \to \texttt{Int}) \\
\mathcal{W}(A_0, \texttt{fix}) & = & (id, (a_1 \to a_1) \to a_1) \\
\mathcal{W}(A_0, \lambda\,\texttt{fact}\,x.\ \texttt{if}\ (x\ \texttt{<=0})\ \texttt{1}\ (\texttt{fact}\,(x - \texttt{1}) * x)) & = & ([\ldots], (\texttt{Int} \to \texttt{Int}) \to (\texttt{Int} \to \texttt{Int})) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1\}, \lambda\,x.\ \texttt{if}\ (x\ \texttt{<=0})\ \texttt{1}\ (\texttt{fact}\,(x - \texttt{1}) * x)) & = & ([b_1/\texttt{Int} \to \texttt{Int}, \ldots], \texttt{Int} \to \texttt{Int}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: b_2\}, \texttt{if}\ (x\ \texttt{<=0})\ \texttt{1}\ (\texttt{fact}\,(x - \texttt{1}) * x)) & = & ([b_2/\texttt{Int}, b_1/\texttt{Int} \to \texttt{Int}, \ldots], \texttt{Int}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: b_2\}, \texttt{if}\ (x\ \texttt{<=0})\ \texttt{1}) & = & ([b_2/\texttt{Int}, \ldots], \texttt{Int} \to \texttt{Int}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: b_2\}, \texttt{if}\ (x\ \texttt{<=0})) & = & ([b_2/\texttt{Int}, \ldots], a_2 \to a_2 \to a_2) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: b_2\}, \texttt{if}) & = & (id, \texttt{Bool} \to a_2 \to a_2 \to a_2) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: b_2\}, (x\ \texttt{<=0})) & = & ([b_2/\texttt{Int}, \ldots], \texttt{Bool}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: b_2\}, (x\ \texttt{<=})) & = & ([b_2/\texttt{Int}, \ldots], \texttt{Int} \to \texttt{Bool}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: b_2\}, \texttt{<=}) & = & (id, \texttt{Int} \to \texttt{Int} \to \texttt{Bool}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: b_2\}, x) & = & (id, b_2) \\
mgu(\texttt{Int} \to \texttt{Int} \to \texttt{Bool}, b_2 \to b_3) & = & [b_2/\texttt{Int}, b_3/\texttt{Int} \to \texttt{Bool}] \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, \texttt{0}) & = & (id, \texttt{Int}) \\
mgu(\texttt{Int} \to \texttt{Bool}, \texttt{Int} \to b_4) & = & [b_4/\texttt{Bool}] \\
mgu(\texttt{Bool} \to a_2 \to a_2 \to a_2, \texttt{Bool} \to b_5) & = & [b_5/a_2 \to a_2 \to a_2] \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, \texttt{1}) & = & (id, \texttt{Int}) \\
mgu(a_2 \to a_2 \to a_2, \texttt{Int} \to b_6) & = & [b_6/\texttt{Int} \to \texttt{Int}] \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, \texttt{fact}\,(x - \texttt{1}) * x) & = & ([b_1/\texttt{Int} \to \texttt{Int}, \ldots], \texttt{Int}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, \texttt{fact}\,(x - \texttt{1}) *) & = & ([b_1/\texttt{Int} \to \texttt{Int}, \ldots], \texttt{Int} \to \texttt{Int}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, *)) & = & (id, \texttt{Int} \to \texttt{Int} \to \texttt{Int}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, \texttt{fact}\,(x - \texttt{1})) & = & ([b_1/\texttt{Int} \to b_9, \ldots], b_9)
\end{array}
$$

$$
\begin{array}{rcl}
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, \texttt{fact}) & = & (id, b_1) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, x - \texttt{1}) & = & ([\ldots], \texttt{Int}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, x-) & = & ([\ldots], \texttt{Int} \to \texttt{Int}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, -) & = & (id, \texttt{Int} \to \texttt{Int} \to \texttt{Int}) \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, x) & = & (id, \texttt{Int}) \\
mgu(\texttt{Int} \to \texttt{Int} \to \texttt{Int}, \texttt{Int} \to b_7) & = & [b_7/\texttt{Int} \to \texttt{Int}] \\
\mathcal{W}(A_0 + \{\texttt{fact} :: b_1, x :: \texttt{Int}\}, \texttt{1}) & = & (id, \texttt{Int}) \\
mgu(\texttt{Int} \to \texttt{Int}, \texttt{Int} \to b_8) & = & [b_8/\texttt{Int}] \\
mgu(b_1, \texttt{Int} \to b_9) & = & [b_1/\texttt{Int} \to b_9] \\
mgu(\texttt{Int} \to \texttt{Int} \to \texttt{Int}, b_9 \to b_{10}) & = & [b_9/\texttt{Int}, b_{10}/\texttt{Int} \to \texttt{Int}] \\
\mathcal{W}(A_0 + \{\texttt{fact} :: \texttt{Int} \to \texttt{Int}, x :: \texttt{Int}\}, x) & = & (id, \texttt{Int}) \\
mgu(\texttt{Int} \to \texttt{Int}, \texttt{Int} \to b_{11}) & = & [b_{11}/\texttt{Int}] \\
mgu(\texttt{Int} \to \texttt{Int}, \texttt{Int} \to b_{12}) & = & [b_{12}/\texttt{Int}] \\
mgu((a_1 \to a_1) \to a_1, ((\texttt{Int} \to \texttt{Int}) \to (\texttt{Int} \to \texttt{Int})) \to b_{13}) & = & [a_1/\texttt{Int} \to \texttt{Int}, b_{13}/\texttt{Int} \to \texttt{Int}]
\end{array}
$$