GADTs Meet Their Match:

Pattern-Matching Warnings That Account for GADTs, Guards, and Laziness

SEBASTIAN GRAF, Karlsruhe Institute of Technology, Germany
SIMON PEYTON JONES, Microsoft Research, UK

Authors' addresses: Sebastian Graf, Karlsruhe Institute of Technology, Karlsruhe, Germany, sebastian.graf@kit.edu; Simon Peyton Jones, Microsoft Research, Cambridge, UK, simonpj@microsoft.com.

Guard Syntax

$$
\begin{array}{rcll}
K \in & \mathsf{Con} & & \\
x, y, a, b \in & \mathsf{Var} & & \\
\tau, \sigma \in & \mathsf{Type} & & \\
e \in & \mathsf{Expr} & ::= & x : \tau \\
& & | & K\ \overline{a}\ \overline{\gamma}\ \overline{e : \tau} \\
& & | & \ldots
\end{array}
\qquad
\begin{array}{rcll}
\gamma \in & \mathsf{TyCt} & ::= & \tau_1 \sim \tau_2 \mid \ldots \\
g \in & \mathsf{Grd} & ::= & \mathsf{let}\ x : \tau = e \\
& & | & K\ \overline{a}\ \overline{\gamma}\ \overline{y : \tau} \leftarrow x \\
& & | & !x
\end{array}
$$

DNF Syntax

$$
\begin{array}{rcll}
\Gamma & ::= & \varnothing \mid \Gamma, x : \tau \mid \Gamma, a & \text{Context} \\
\delta & ::= & \gamma \mid K\ \overline{x : \tau} \leftarrow y \mid x \not\approx K \mid x \approx \bot \mid x \not\approx \bot \mid x \approx e & \text{Literals} \\
\Delta & ::= & \checkmark \mid \Delta \wedge \delta & \text{Clause} \\
\nabla & ::= & \times \mid \Delta \mid \nabla_1 \vee \nabla_2 & \text{Formula}
\end{array}
$$

$$
\begin{array}{lcl}
\times & \oplus \delta & = \times \\
\Delta & \oplus \delta & = \Delta \wedge \delta \\
\nabla_1 \vee \nabla_2 & \oplus \delta & = (\nabla_1 \oplus \delta) \vee (\nabla_2 \oplus \delta)
\end{array}
$$

Clause Tree Syntax

$$
\begin{array}{rcl}
\mathcal{T}[r] & ::= & \mathtt{Rhs} \mid \mathtt{Many}\ \overline{r} \\
t_G \in \mathsf{Gdt} & ::= & \mathcal{T}[t_G] \mid \mathtt{Guard}\ g\ t_G \\
t_A \in \mathsf{Ant} & ::= & \mathcal{T}[t_A] \mid \mathtt{MayDiverge}\ t_A \mid \mathtt{Inaccessible}\ t_A
\end{array}
$$

Checking Guard Trees

$$\boxed{\mathcal{U}(\nabla, \mathsf{Gdt}) = \nabla}$$

$$
\begin{array}{lcl}
\mathcal{U}(\nabla, \mathtt{Rhs}) & = & \times \\
\mathcal{U}(\nabla, \mathtt{Many}\ \overline{t}) & = & \mathcal{U}(\ldots \mathcal{U}(\mathcal{U}(\nabla, t_1), t_2) \ldots, t_n) \\
\mathcal{U}(\nabla, \mathtt{Guard}\ (!x)\ t) & = & \mathcal{U}(\nabla \oplus (x \not\approx \bot), t) \\
\mathcal{U}(\nabla, \mathtt{Guard}\ (\mathsf{let}\ x = e)\ t) & = & \mathcal{U}(\nabla \oplus (x \approx e), t) \\
\mathcal{U}(\nabla, \mathtt{Guard}\ (K\ \overline{a}\ \overline{\gamma}\ \overline{y : \tau} \leftarrow x)\ t) & = & (\nabla \oplus (x \not\approx K) \oplus (x \not\approx \bot)) \cup \mathcal{U}(\nabla \oplus (K\ \overline{y : \tau} \leftarrow x) \oplus \overline{\gamma}, gs)
\end{array}
$$

$$\boxed{\mathcal{A}_\Gamma(\nabla, \mathsf{Gdt}) = \mathsf{Ant}}$$

$$
\begin{array}{lcl}
\mathcal{A}_\Gamma(\nabla, \mathtt{Rhs}) & = & \begin{cases} \mathtt{Inaccessible\ Rhs}, & \mathcal{V}(\Gamma, \nabla) = \varnothing \\ \mathtt{Rhs}, & \text{otherwise} \end{cases} \\
\mathcal{A}_\Gamma(\nabla, \mathtt{Many}\ \overline{t}) & = & \mathtt{Many}\ (\mathcal{A}_\Gamma(\nabla'_0, t_1), \ldots, \mathcal{A}_\Gamma(\nabla'_{n-1}, t_n)) \text{ where } \begin{cases} \nabla'_0 & = \nabla \\ \nabla'_{n+1} & = \mathcal{U}(\nabla'_n, t_{n+1}) \end{cases} \\
\mathcal{A}_\Gamma(\nabla, \mathtt{Guard}\ (!x)\ t) & = & \mathcal{D}(\nabla, \mathcal{A}_\Gamma(\nabla \oplus (x \not\approx \bot), t)) \\
\mathcal{A}_\Gamma(\nabla, \mathtt{Guard}\ (\mathsf{let}\ x = e)\ t) & = & \mathcal{A}_\Gamma(\nabla \oplus (x \approx e), t) \\
\mathcal{A}_\Gamma(\nabla, \mathtt{Guard}\ (K\ \overline{a}\ \overline{\gamma}\ \overline{y : \tau} \leftarrow x)\ t) & = & \mathcal{D}(\nabla, \mathcal{A}_\Gamma(\nabla \oplus (K\ \overline{y : \tau} \leftarrow x) \oplus \overline{\gamma}, t))
\end{array}
$$

$$\boxed{\mathcal{D}(\nabla, \mathsf{Ant}) = \mathsf{Ant}}$$

$$
\mathcal{D}(\nabla, t) = \begin{cases} t, & \mathcal{V}(\Gamma, \nabla \oplus (x \approx \bot)) = \varnothing \\ \mathtt{MayDiverge}\ t & \text{otherwise} \end{cases}
$$

$$\boxed{\mathcal{V}(\Gamma, \nabla) = \mathit{values}}$$

TBD: Oracle implementation. This is `provideEvidence`

Putting it all together

(0) Input: Context with match vars Γ and desugared Gdt t
(1) Report n value vectors of $\mathcal{V}(\Gamma, \mathcal{U}(\checkmark, t))$ as uncovered
(2) Report the collected redundant and not-redundant-but-inaccessible clauses in $\mathcal{A}_\Gamma(\checkmark, t)$ (TODO: Write a function that collects the RHSs, maybe add numbers to `Rhs` to distinguish).

This figure is completely out of date, don't waste your time

Test if Oracle state Delta is unsatisfiable

$$\frac{\boxed{\nvdash_{\text{SAT}} \Gamma \vdash \Delta}}{\nvdash_{\text{SAT}} \Gamma \vdash fvs\Gamma \triangleright \Delta}$$

Test a list of SAT roots for inhabitants

$$\frac{\boxed{\nvdash_{\text{SAT}} \Gamma \vdash \overline{x} \triangleright \Delta}}{\nvdash_{\text{SAT}} \Gamma \vdash x_i \triangleright \Delta}$$

Test a single SAT root for inhabitants

$$\frac{\boxed{\nvdash_{\text{SAT}} \Gamma \vdash x \triangleright \Delta}}{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta x \approx \bot \quad \{\overline{K}\} \text{ COMPLETE set} \quad \overline{\forall \overline{y : \tau}.\, \nvdash_{\text{SAT}} \Gamma, \overline{y : \tau} \vdash \oplus\Delta x \approx K\ \overline{y}}}$$

Add a single equality to Δ

$$\boxed{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta\delta}$$

Term stuff: Bottom, negative info, positive info + generativity, positive info + univalence

$$\frac{x \not\approx sth \in \Delta}{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta x \approx \bot} \qquad \frac{x \approx K\ \overline{y} \in \Delta}{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta x \approx \bot}$$

$$\frac{x \not\approx K \in \Delta}{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta x \approx K\ \overline{y}} \qquad \frac{x \approx K_i\ \overline{y} \in \Delta \quad i \neq j \quad K_i \text{ and } K_j \text{ generative}}{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta x \approx K_j\ \overline{z}}$$

$$\frac{x \approx K\ \overline{\tau}\ \overline{y} \in \Delta \quad \nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta\tau_i \sim \sigma_i}{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta x \approx K\ \overline{\sigma}\ \overline{z}} \qquad \frac{x \approx K\ \overline{\tau}\ \overline{y} \in \Delta \quad \nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta y_i \approx z_i}{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta x \approx K\ \overline{\sigma}\ \overline{z}}$$

Type stuff: Hand over to unspecified type oracle

$$\frac{\tau_1 \text{ and } \tau_2 \text{ incompatible to Givens in } \Delta \text{ according to type oracle}}{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta\tau_1 \sim \tau_2}$$

Mixed: Instantiate K and see if that leads to a contradiction TODO: Proper instantiation

$$\frac{\overline{\nvdash_{\text{SAT}} \Gamma \vdash y \triangleright \Delta \cup y \not\approx \bot}}{\nvdash_{\text{SAT}} \Gamma \vdash \oplus\Delta x \approx K\ \overline{y}}$$