CBOR as language runtime support

• Many language environment need to store and interchange structured data
• CBOR has the right support for complex types in many languages
• Some tags already defined for this, e.g., tag 1001
• Many complex types are defined by specific programs
  • Some support already in, e.g., tag 27 (originally motivated by Perl)
• Gap: **discriminated unions**
Union types

- Many type systems and schema languages allow specific items to be one of multiple types
  - Cf. CDDL choices, foo = a / b / c
  - In many cases, an actual **union** (all values from each of the choices) is needed; easy to express in CBOR
  - In other cases, alternatives look **structurally** the same in terms of CBOR data items, but still need to be distinguished
    - Add **discriminator** (often a map label)
    - Don’t want to force discriminated unions into a specific structure, though
Example

data Expr = Lit Int          -- integer literal
  | Add Expr Expr           -- addition
  | Sub Expr Expr           -- subtraction
  | Neg Expr                -- unary negation
  | Mul Expr Expr           -- multiplication
  | Div Expr Expr           — integer division

• Here, Add, Sub etc. stand for different choices, but the data look the same
Compiler translation

\[ \text{expr} = \text{Tag0}(\text{int}) \quad ; \quad \text{integer literal} \]
\[ / \quad \text{Tag1}([\text{expr}, \text{expr}]) \quad ; \quad \text{addition} \]
\[ / \quad \text{Tag2}([\text{expr}, \text{expr}]) \quad ; \quad \text{subtraction} \]
\[ / \quad \text{Tag3}(\text{expr}) \quad ; \quad \text{unary negation} \]
\[ / \quad \text{Tag4}([\text{expr}, \text{expr}]) \quad ; \quad \text{multiplication} \]
\[ / \quad \text{Tag5}([\text{expr}, \text{expr}]) \quad ; \quad \text{integer division} \]

- Define tags that have a local meaning within a specific discriminated union
- Don’t need to register global “Add” tag
Proposal: register generic discriminators

expr = 185(int) ; integer literal
    / 186([expr, expr]) ; addition
    / 187([expr, expr]) ; subtraction
    / 188(expr) ; unary negation
    / 189([expr, expr]) ; multiplication
    / 190([expr, expr]) ; integer division

• ➔ Register 7 1+1, ~2048 1+2, and a catch-all 1+1+array